Huffman 부호와 평균부호길이 함수의 특성을 이용한 양방향 가변길이 부호의 생성 방법

정 욱 현, 윤 영 석, 호 요 성
광주과학기술원 정보 통신 공학과
전화 : 062-970-2258 / 핸드폰 : 019-9183-3542

Design of Reversible Variable-Length Codes Using Properties of the Huffman Code and the Average Length Function

Wook-Hyun Jeong, Young-Suk Yoon, and Yo-Sung Ho
Kwangju Institute of Science and Technology (K-JIST)
E-mail : {whjeong, ysyoon, hoyo}@kjist.ac.kr

Abstract

In this paper, we propose a new construction algorithm for the reversible variable-length code (RVLC) using a simplified average length function of the optimal Huffman code. RVLC is introduced as one of the error resilience tools in H.263+ and MPEG-4 owing to its error-correcting capability. The proposed algorithm demonstrates an improved performance in terms of the average codeword length over the existing RVLC algorithms.

I. 서론

일반적으로 Huffman 부호[1]나 산술 부호[2]와 같은 가변길이 부호를 사용하여 압축된 비디오 비트열(bitstream)은 비트 오류에 매우 취약하다. 가변길이 부호를 전송 오류에 강인하도록 설계한 양방향 가변길이 부호(reversible variable-length code, RVLC)는 순방향과 역방향 모두 복호가 가능한 부호로써 어두(prefix) 조건 뿐 아니라, 어미(suffix) 조건도 만족하는 특성이 있다. RVLC의 사용은 전송 오류가 발생했을 때, 양방향 복호를 통해 전송된 비트열로부터 손상되지 않은 데이터를 가능한 한 많이 복원할 수 있다는 장점을 가진다. RVLC는 부호어 구성의 모양에 따라서 대칭적(symmetrical) RVLC와 비대칭적(asymmetrical) RVLC로 나뉘며[5], 대칭적 RVLC와 비대칭적 RVLC는 각각 H.263+와 MPEG-4에 채택되었다[3, 4].

Takishima[5]는 주어진 Huffman 부호에 기초하여 평균부호길이가 엔트로피에 가깝도록 설계된 RVLC 생성 알고리즘을 처음으로 제안하였고, Tsai[6]는 이 알고리즘을 개선하여 보다 짧은 평균부호길이를 갖는 알고리즘을 제안하였다. Tseng[7]은 Huffman 부호를 이용하지 않고 대칭적 RVLC를 설계하는 방법을 제안하였다. 이 방식은 부호어가 심볼(symbol)에 할당될 때 발생 확률에 부호어의 비트 길이를 곱하여 합한 값을 작게 만드는 부호어를 찾는 일종의 과도(exhaustive) 검색 방법이다. Lin[8]은 비대칭적 RVLC의 생성 알고리즘에 Tseng의 방법을 적용하였다.

하지만 이러한 알고리즘들을 분석하면 부호화 효율성 측면에서 몇 가지 개선의 여지를 보여준다. Huffman 부호 기반의 방법들은 주어진 허프만 부호에 대한 적응 과정에 제한 사항이 있어 사용 가능한 RVLC를 놓치는 경우가 발생한다. Huffman 부호 기반이 아닌 방법들은 기존의 Huffman 부호 기반의 방법들보다 효율적인 RVLC를 제공하지만 부호 설계에 있어 필수적인 시작 비트 길이와 각 레벨에서의 부호 선택 기법 등이 불명확하다.

본 논문에서는 Huffman 부호를 기반으로, 평균부호길이를 비교해 가며 각 레벨에서 보다 효율적인 부호어들을 검색하는 방식을 사용하여 어떠한 형태의 발생확률 분포를 갖는 소스(source)에 대해서도 보다 효율적인 RVLC를 생성할 수 있는 RVLC의 설계 알고리즘을 제안한다. 제안한 방법은 주어진 Huffman 부호로부터 얻을 수 있는 중요한 정보와 평균부호길이 함수의 특성을 이용하여 효율적인 RVLC가 존재하는 영역을 비트 레벨 단계에서 구체화시킨다.

II. Huffman 부호의 특성

기존의 알고리즘들처럼 제안된 RVLC 생성 방식은 하향(top-down) 방식을 따른다. 즉, Huffman의 방법[1]처럼 발생확률이 가장 작은 심볼에서 출발하지 않고, 가장 빈번하게 발생하는 심볼부터 짧은 비트를 할당하며, 전체 심볼수 S만큼 모든 부호어가 할당될 때까지 생성과정을 진행한다. 하향 방식으로 주어진 소스에 대하여 최적인 비대칭적 RVLC를 생성하려면 가능한 모든 경우의 RVLC를 생성한 후 가장 짧은 평균부호길이를 갖는 RVLC를 선택하는 것이 확실한 방법이다. 하지만 하향 방식을 통해 설계 가능한 대칭적 혹은 비대칭적 RVLC의 수는 무한하다.

하향 방식에서 최종 대칭적비대칭적 RVLC를 구성하는 각 부호어의 결정은 다음의 몇 가지 요소, RVLC의 가장 짧은 비트 길이 레벨(level)과 각 비트 길이에 존재하는 부호어 수 및 해당 부호어에 따른다. 하지만 Huffman 부호의 부호어 구성 방식을 적용하면 RVLC의 결정 요소를 조절하여 보다 효율적인 RVLC가 존재하는 영역을 구체화시킬 수 있다.

최적화된 Huffman 부호에서 가장 발생확률이 높은 심볼에 가장 짧은 비트 길이, L_{min}이 할당되고, 이는 심볼의 수와 발생확률 분포 등에 영향을 받는다. 그리고, L_{min}은 주어진 발생확률에서 최적의 부호를 설계하기 위한 필수적인 요소이다.

비대칭적 RVLC는 구성되는 부호어들의 비트 패턴(pattern)을 구분하지 않아서 부호어 할당에 있어 Huffman 부호와 비슷한 구성을 보인다. 따라서, 우리는 비대칭적 RVLC를 설계하기 위해 RVLC의 시작 비트 레벨로서 L_{min}을 사용한다. 대칭적 RVLC는 실험을 통해 근사적으로 균등한 확률분포를 갖는 소스일 경우 L_{min}과 (L_{min}-1)에서 보다 짧은 평균부호길이를 보인다. 반대로 수 개의 심볼에 발생확률이 집중된 경우에는 L_{min}과 (L_{min}+1)에서 더 나은 부호화 성능을 보인다. 따라서, 대칭적 RVLC를 설계할 경우 주어진 발생확률 분포에 따라서 L_{min}과 더불어 (L_{min}-1), (L_{min}+1)에 대해 고려해야 한다.

비트 길이 벡터(vector) $n(i)$를 비트 길이 i를 갖는 부호어의 수라고 정의한다. 그리고, $n_{Huff}(i)$와 $n_{RVLC}(i)$를 각각 주어진 Huffman 부호와 RVLC의 비트 길이 벡터로 정의한다. 제안된 알고리즘도 과도 검색을 사용하여 L_{min}과 그 하위 레벨에서 선택해야 하는 부호어와 그 수, $n_{RVLC}(i)$를 결정한다. 그리고, 이를 결정할 때에 주어진 Huffman 부호의 $n_{Huff}(L_{min})$과 부호 구성 방식, 그리고 Z_L[9]을 이용하여 최적의 RVLC로 접근할 수 있는 경로를 찾는다. 각 레벨에서 존재할 수 있는 $n_{RVLC}(i)$는 어미 조건에 의해 제한되므로 최상의 레벨에서의 $n_{RVLC}(i)$는 $n_{Huff}(L_{min})$와 같거나 많아야 RVLC의 효율성을 증가시킬 수 있다. 따라서 검색의 시작 지점을 $n_{Huff}(L_{min})$로 결정한다.

주어진 발생 확률에서 Laplacian 분포와 같이 몇 개의 심볼의 확률이 매우 높을 경우에 Huffman 부호의 비트 레벨을 보면 L_{min}에서 출발하여 특정 레벨에서 부호어들이 할당되지 않고 스킵(skip) 되는 사실을 알 수 있다. 하향 방식으로 효율적인 RVLC를 설계하려면 상위 레벨에서는 적절한 수의 부호어를 할당하면서 동시에 하위 레벨에 많은 부호어를 보장할 수 있어야 하고 하위 레벨에서는 보다 짧은 부호어들을 많이 할당해야 평균부호길이를 줄일 수 있다. 이를 이용하여 제안된 알고리즘에서는 주어진 Huffman 부호에서 $\lceil S/2 \rceil$ 개의 심볼을 포함하는 상위 레벨에서 레벨 스킵이 발생하면 RVLC에서도 해당 레벨은 사용하지 않는다. 여기서 $\lceil x \rceil$는 x보다 크거나 같은 수 중에 가장 작은 정수를 말한다. 하위 레벨의 부호어들은 레벨 스킵과 상관없이 어두 (prefix) 조건과 어미(suffix) 조건을 동시에 만족하는 부호어들로 구성된다.

Z_L은 L 비트 길이의 모두 '0'비트로만 이루어진 부호어를 말한다. 이 부호어는 모두 '1'비트로만 이루어진 부호어와 같이 전체 RVLC에서 한 번은 반드시 선택되는 부호어이므로 최상위 레벨인 L_{min}에 할당하여 효율성을 개선하는 데에 이용한다.

III. 평균부호길이 함수의 특성

부호화 효율을 측정하기 위해 우리는 평균부호길이를 사용한다. 그리고, 평균부호길이를 특정 레벨에서 부호어 수에 대한 함수로 보면 평균부호길이 함수는 구간 내에서 최소값을 갖는 볼록(convex) 함수이며 이 최소값을 기준으로 부호어 수에 따라 단조 증가 혹은 단조 감소 함수이다. 예를 들어 1부터 $2^{L_{min}}$까지의 범위를 갖는 $n_{RVLC}(L_{min})$에 대하여 평균부호길이 함수는 [1, $2^{L_{min}}$]에서 최소값을 가진다. 평균부호길이는 볼록 함수이므로 국부적인(local) 최소값은 전체적인(global) 최소값을 대표하므로 보다 효율적인 RVLC의 $n_{RVLC}(L_{min})$은 이 최소값을 갖는 지점에 위치한다. 대칭적 RVLC의 경우에는 주어진 Huffman 부호에서 레벨 스킵의 발생 여부에 따라 상위 레벨에 레벨 스킵이 발생하였다면 (L_{min}+1)을, 그렇지 않으면 (L_{min}-1)에 대한 경우를 고려해야 한다.

$\lceil S/2 \rceil$를 포함하는 L_{min}을 제외한 상위 레벨에서 각 레벨의 부호어 수 $n_{RVLC}(i)$, ($i > L_{min}$)를 결정하는 방식은 다음과 같다.

(a) 이전 비트 레벨 (L_{curr}-1)까지 선택된 RVLC에 의하여 현재 레벨 L_{curr}에서 사용 가능한 RVLC가 결정

되고 이 부호어의 총수를 $n_{RVLC}(L_{curr})$로 놓는다.

(b) $(n_{RVLC}(L_{curr})-l,\ 0 \le l < n_{RVLC}(L_{curr}))$에서 평균부호길이가 최소값을 가지면 $n_{RVLC}(L_{curr})$는 다음과 같이 대치된다. 여기에서 l은 0부터 '1'씩 증가된다.

$$n_{RVLC}(L_{curr}) = n_{RVLC}(L_{curr})-l \qquad (1)$$

여기에서 $\begin{pmatrix} n_{RVLC}(L_{curr}) \\ n_{RVLC}(L_{curr})-l \end{pmatrix}$만큼의 RVLC가 생성되고 평균부호길이를 비교하게 된다. 여기서 $\begin{pmatrix} a \\ b \end{pmatrix}$는 a개에서 b만큼을 선택하는 조합수이다.

(c) 평균부호길이는 볼록 함수이므로 현재 $n_{RVLC}(L_{curr})$에서 최소의 평균부호길이가 이전 값보다 크다면 이전의 $n_{RVLC}(L_{curr})$가 선택되고 그 때의 부호어들이 L_{curr}에서 사용 가능한 부호어들이 된다.

제안된 RVLC의 생성과정을 정리하면 다음과 같다.

1) $n_{Huff}(L_{min})$에서 출발하여 RVLC를 생성하고 평균부호길이를 비교하여 $n_{RVLC}(L_{min})$을 결정한다. RVLC를 생성할 때에 주어진 Huffman 부호에서 S의 반수를 포함하는 상위 레벨에 스킵된 레벨은 제외한다. 하나의 부호어는 반드시 Z_L을 선택한다. 대칭적 RVLC이고 스킵된 레벨이 없다면 $(L_{min}-1)$에 대하여 Z_{L-1}이 적용된다. 반대로 레벨 스킵이 발생하였으면 $(L_{min}-1)$에 대하여 Z_{L+1}이 적용된다.
2) $\lceil S/2 \rceil$를 포함하는 상위 레벨에 대하여 각 레벨의 부호어를 선택하는 과정이 동작된다.
3) S에 대한 모든 부호어가 할당될 때까지 하위 레벨에서 어두 (prefix) 조건과 어미(suffix) 조건을 동시에 만족하는 부호어들을 선택하면 최종 RVLC를 얻을 수 있다.

IV. 실험 결과

표 1은 기존의 방법들과 제안된 방법을 가지고 압축 알고리즘에 대한 성능 평가를 위해 사용되는 Canterbury 파일(http://corpus.canterbury.ac.nz)에 대하여 대칭적비대칭적 RVLC를 생성하여 평균부호길이로써 부호화 효율을 비교한 결과를 보여준다. 제안된 RVLC는 대칭적 RVLC의 경우 Tseng[7]의 RVLC와 동일한 효율을 보이고 비대칭적 RVLC의 경우 기존의 Lin[8]의 RVLC보다 훨씬 줄어든 평균부호길이를 가진다. 특히 11개의 파일 중 F1, F3, F5, F6, F7을 제외한 나머지 파일들은 레벨 스킵이 적용되었고 기존의 알고리즘과 현저한 효율성의 차이를 보인다.

표 2는 기존의 방법들과 제안된 방법을 통해 영어 알파벳에 대하여 대칭적비대칭적 RVLC를 생성하여 부호어 할당과 그에 따른 평균부호길이를 보여준다. 알파벳의 경우 레벨 스킵은 발생하지 않았으며 대칭적 RVLC의 경우 역시 Tseng[7]의 부호와 동일하다. 비대칭적 RVLC의 경우 Lin[8]의 RVLC보다 효율성이 0.34% 정도 개선된 결과를 보이고 있다.

V. 결론

본 논문에서는 최적의 Huffman 부호를 기초하고, 평균부호길이를 하나의 레벨에 존재하는 부호어 수에 대한 함수로 만들어 대칭적비대칭적 RVLC를 생성하는 새로운 알고리즘을 제안하였다. 주어진 소스에 대하여 하향 방식을 이용하여 생성된 RVLC는 무수히 많을 수 있지만, 제안된 알고리즘은 주어진 Huffman 부호로부터 얻는 결정적인 특성들, 즉 L_{min}과 상위 레벨에서 스킵된 레벨을 적용시키고 평균부호길이 함수의 특성을 이용하여 각 레벨에서 사용 가능한 부호어들을

표 1. 다양한 소스 파일에 대하여 기존의 알고리즘과 제안된 알고리즘을 사용하여 생성된 대칭적비대칭적 RVLC의 평균부호길이 비교 (▭ 레벨 스킵)

File		부호어 수 (심볼 수)	Huffman 부호	대칭적 RVLC			비대칭적 RVLC		
				Tsai의 방법	Tseng의 방법	제안된 방법	Tsai의 방법	Lin의 방법	제안된 방법
			평균부호길이	평균부호길이	평균부호길이	평균부호길이	평균부호길이	평균부호길이	평균부호길이
F1	asvoulik.txt	68	4.84465	5.27886	5.21025	5.21025	5.01142	5.00954	4.85262
F2	alice29.txt	74	4.61244	5.01398	4.93155	4.93155	4.80326	4.68871	4.68856
F3	xargs.1	74	4.92382	5.39863	5.33996	5.33996	5.07334	5.16087	4.93577
F4	grammar.lsp	76	4.66434	5.04757	5.01774	5.01774	4.85461	4.78581	4.72571
F5	plabn12.txt	81	4.57534	4.94473	4.89527	4.89527	4.80659	4.64910	4.63477
F6	lcet10.txt	84	4.69712	5.12232	5.01682	5.01682	4.87868	4.74177	4.72741
F7	cp.html	86	5.26716	5.85839	5.81173	5.81173	5.37113	5.77080	5.28112
F8	fields.c	90	5.04090	5.47596	5.46332	5.46332	5.26987	5.20278	5.08843
F9	ptt5	159	1.66091	1.77735	1.75992	1.75992	1.71814	1.70401	1.67630
F10	Sum	255	5.36504	6.10683	6.03917	6.03917	5.49767	6.01870	5.41683
F11	kennedy.xls	256	3.59337	4.25681	4.21092	4.21092	3.89401	3.85384	3.78418

표 2. 영어 알파벳에 대하여 기존의 알고리즘과 제안된 알고리즘을 사용하여 생성된 대칭적비대칭적 RVLC

발생확률		Huffman 부호		대칭적 RVLC								비대칭적 RVLC							
				Takishima의 방법		Tsai의 방법		Tseng의 방법		제안된 방법		Takishiam의 방법		Tsai의 방법		Lin의 방법		제안된 방법	
		L	부호어	L	부호어	L	부호어	L	부호어	L	부호어	L	부호어	L	부호어	L	부호어	L	부호어
E	0.14878570	3	001	3	000	3	010	3	000	3	000	3	001	3	000	3	000	3	000
T	0.09354149	3	110	3	111	3	101	3	111	3	111	3	110	3	111	3	100	3	101
A	0.08833733	4	0000	4	0110	4	0110	3	010	3	010	4	0000	4	0101	3	101	3	110
O	0.07245796	4	0100	4	1001	4	1001	3	101	3	101	4	0100	4	1010	4	0010	4	0010
R	0.06872164	4	0101	5	00100	4	0000	4	0110	4	0110	4	0101	4	0010	4	0011	4	0011
N	0.06498532	4	0110	5	11011	4	1111	4	1001	4	1001	4	1000	4	1101	4	0110	4	0111
H	0.05831331	4	1000	5	01010	5	01110	5	00100	5	00100	4	1010	4	0100	4	0111	4	1001
I	0.05644515	4	1001	5	10101	5	10001	5	11011	5	11011	5	10010	4	1011	4	1110	4	0100
S	0.05537763	4	1010	5	01110	5	00100	5	01110	5	01110	5	01100	4	0110	4	1111	4	1111
D	0.04376834	5	00010	5	10001	5	11011	5	10001	5	10001	5	00010	5	11001	5	01001	5	01010
L	0.04123298	5	00011	6	001100	6	011110	6	001100	6	001100	5	00011	5	10011	5	01010	5	01011
U	0.02762209	5	10110	6	110011	6	100001	6	110011	6	110011	5	10111	5	01110	5	01011	5	01100
P	0.02575393	5	10111	6	010010	6	001100	6	011110	6	011110	5	11100	5	10001	5	11001	5	10001
F	0.02455297	5	11100	6	101101	6	110011	6	100001	6	100001	5	11111	6	001100	5	11010	5	11100
M	0.02361889	5	11110	6	011110	7	0111110	7	0010100	7	0010100	6	111101	6	011110	5	11011	6	011010
C	0.02081666	5	11111	6	100001	7	1000001	7	1101011	7	1101011	6	101101	6	100001	6	010001	6	011011
W	0.01868161	6	011100	7	0010100	7	0010100	7	0011100	7	0011100	6	011101	7	1001001	6	110001	6	100001
G	0.01521216	6	011101	7	1101011	7	1101011	7	1100011	7	1100011	6	111011	7	0011100	7	0100001	6	111010
Y	0.01521216	6	011110	7	0011100	7	0011100	7	0111110	7	0111110	8	01110011	7	1100011	7	1100001	6	111011
B	0.01267680	6	011111	7	1100011	7	1100011	7	1000001	7	1000001	8	11101011	7	0111110	8	01000001	7	1000001
V	0.01160928	6	111011	7	0100010	7	0001000	8	00111100	8	00111100	9	111010011	7	1000001	8	11000001	8	10000001
K	0.00867360	7	1110100	7	1011101	7	1110111	8	11000011	8	11000011	9	011110011	8	00111100	9	010000001	9	100000001
X	0.00146784	8	11101011	8	00111100	8	01111110	8	01111110	8	01111110	10	0111110011	8	11000011	9	110000001	10	1000000001
J	0.00080064	9	111010101	9	001010100	9	011111110	8	10000001	8	10000001	10	1110101011	9	100101001	10	0100000001	11	10000000001
Q	0.00080064	10	1110101000	10	0010110100	10	0111111110	9	011111110	9	011111110	11	11101010011	10	00111101001	10	1100000001	12	100000000001
Z	0.00053376	10	1110101001	10	1101001011	10	1000000001	9	100000001	9	100000001	13	1110101000111	10	1001011100	11	01000000001	13	1000000000001
평균부호길이		4.15572392		4.65695649		4.60728507		4.46463681		4.46463681		4.42607344		4.30677804		4.1834808		4.172804	

선택하며 효율적인 RVLC를 점차적으로 설계한다. 실험 결과를 통해 제안된 기법이 기존의 설계 기법보다 더 우수한 효율을 갖는 RVLC 부호를 생성하는 사실을 확인할 수 있었다.

감사의 글

본 연구는 광주과학기술원(K-JIST)과 광주과학기술원 실감방송 연구센터를 통한 대학IT연구센터(ITRC), 그리고 교육부 두뇌한국21 (BK21) 정보기술사업단의 지원에 의한 것입니다.

참고문헌

[1] D. Huffman, "A method for the construction of minimum redundancy codes," *Proc. Inst. Radio Engr.*, vol. 40, pp. 1098-1101, Sept. 1952.

[2] J.J. Rissanen and G.G. Langdon. Jr., "Arithmetic coding," *IBM J. Res. Develop.*, 23, pp. 149-162, 1979.

[3] ITU-T Rec. H.263, "Video coding for low bit rate communication," Annex V, 2000.

[4] ISO/IEC 14496-2, "Information technology - coding of audio-visual objects," *Final Draft Int. Std.*, Part 2 : Visual, Oct. 1998.

[5] Y. Takishima, M. Wada, and H. Murakami, "Reversible variable length codes," *IEEE Trans. Comm.*, vol. 43, pp.158-162, Fed. 1995.

[6] C. W. Tsai and J. L. Wu, "On constructing the Huffman-code-based reversible variable length codes," *IEEE Trans. Comm.*, vol. 49, pp. 1506-1509, Sept. 2001.

[7] H.W. Tseng and C.C. Chang, "Construction of symmetrical reversible variable length codes using back-tracking," *Comp. J.*, vol. 46, no. 1, Jan. 2003.

[8] C.W. Lin, Y.J. Chuang, and J.L. Wu, "Generic construction algorithms for symmetric and asymmetric RVLCs," *Proc. IEEE Int. Conf. Communication Systems*, vol. 2, pp. 968-972, Nov. 2002.

[9] W.H. Jeong and Y.S. Ho, "Design of symmetrical reversible variable-length codes from the Huffman code," *Picture Coding Symposium*, pp. 135-138, April 2003.