通信情報保護學會論文誌
第4卷 第2號, 1994. 12

오류제어기능을 갖는 정보원 부호화에 관한 고찰[1)]

이 선 영*, 박 지 환*

A Note on Source Coding Capable of Error Control

Sun-Young LEE and Ji-Hwan PARK

요 약

본 논문에서는 오류제어기능을 갖는 정보원 부호화 방식 중 Fibonacci 부호와 flag 부호에 대하여 알아보고, 부호화와 복호가 간단한 flag 부호에 있어서 비트 삽입 발생에 주목하여 최적의 suffix 패턴을 구한다. 이 최적 suffix 패턴을 사용한 제안부호의 성능을 비트 삽입 발생횟수의 확률적 해석과 컴퓨터 시뮬레이션을 통하여 비교 분석한다. 그 결과, 제안부호의 성능이 평균부호길이면에서 가장 우수한 Fibonacci 부호에 비해 동등이상의 결과를 얻을 수 있음을 보인다.

Abstract

This paper deals with synchronizable codes: Fibonacci code and flag code. We suggest optimal suffix pattern which the number of bit stuffing is minimal as for flag code that is very simple scheme for the positive integer representations.

The performance of the proposed code is evaluated with the probabilistic analysis of bit stuffing and computer simulation. As a result, we can obtain that our code's average code length is shorter than or equal to Fibonacci code which is known the best synchronizable code a view of average code length.

Ⅰ. 서 론

최근 컴퓨터와 통신기술이 발달함에 따라 다루어야 할 데이타량이 급격히 증가하여 데이타 압축의 필요성이 높아지고 있다. 데이타 압축 방법 중 정보원의 통계적 성질을 전제로 하지 않고도 점근적 최량성이 보장되는 유니버셜 부호(universal code)가 많이 연구되어 LZW 부호와 산술부호 등

1) 이 논문은 1993년도 한국학술진흥재단의 공모과제 연구비에 의하여 연구되었음

* 부산수산대학교 전자계산학과

이 실용화되어 널리 응용되고 있다[1]. 이러한 데이타 압축을 위한 정보원 부호화(source coding)는 송신측에서 부호화한 데이타가 수신측에서 오류없이 복호가능함을 가정하고 있다. 그러나, 부호어(codeword) 내에서 오류가 발생했을 때는 압축효과가 높은 부호일수록 오류전파(error propagation)가 커지는 경향이 있어 오류 검출 및 정정 부호에 의한 부가적 수법에 의존하게 된다. 이와 같이 종래에는 정보원 부호화와 통신로 부호화(channel coding)가 각기 독립적으로 수행되어 왔으나, 최근에 정보원 부호화 단계에서 오류를 제어할 수 있는 자기동기 부호(self-synchronizing code)에 관한 연구가 주목되고 있다[2].

유니버셜 부호로 대표되는 적응사전 부호화 방식(dictionary coding)에서는 각 정보원 계열을 일정 규칙에 따라 자연수로 변환하여 2진 부호화할 때, 자연수의 값이 작을수록 발생확률이 높으며, 그 부호어 길이를 짧게 할당하여 압축의 효과를 얻게 된다. 즉, 자연수 $n \in N^+ = \{1,2,\cdots\}$의 확률분포가

$$\sum_{n=1}^{\infty} P(n) = 1$$
$$P(n) \geq P(n+1) \qquad (1)$$

을 만족하고 그 부호어 길이가

$$L(n) \leq L(n+1) \qquad (2)$$

과 같이 최소 조건(minimal property)을 만족하는 경우에 대하여 Elias는 다음과 같은 정의를 사용하였다[3]. 자연수 표현법 ε의 평균 부호 길이 $E_p(L)$이

$$\frac{E_p(L)}{\max\{1, H(P)\}} \leq \rho \qquad (3)$$

을 만족할 때 ε은 유니버셜이라 한다. 이때, H(P)는 P의 분포를 따르는 정보원의 엔트로피이고, ρ는 P와 독립인 임의의 상수이다. 또한, 유니버셜이면서 식(4)를 만족하면 점근적으로 최량(asymptotically optimal)이라고 정의하였다.

$$\frac{E_p(L)}{\max\{1, H(P)\}} \leq R(H(P)) \qquad (4)$$

단, R는 $\lim_{x \to \infty} R(x) = 1$

식(4)는 만족하지 않지만, 어떤 파라미터 f에 대하여

$$\frac{E_p(L)}{\max\{1, H(P)\}} \leq R(H(P), f) \qquad (5)$$

단, R는 $\lim_{f \to \infty} \lim_{x \to \infty} R(x, f) = 1$

을 만족할 때 ε는 거의 점근적으로 최량(almost asymptotically optimal)임을 정의하였다[4].

자연수 $n \in N^+$을 표현하는 부호화 방식은 크게 아래 두가지로 분류된다[8].

i) $n \in N^+$의 2진표현을 B(n)이라 할 때 B(n)의 길이를 prefix로 나타내는 방법[3]

ii) 각 부호어 끝에 suffix를 부가하는 방법[4][5][6].

i)로 분류되는 Elias의 δ, γ 부호는 부호어 내의 임의의 비트에서 오류가 발생하면 재동기(resynchronization)의 간격이 길어지거나 불가능하게 되는 반면에 ii)에 해당되는 부호화 방식들은 부호어내의 suffix 패턴에 의해 동기를 빨리 회복할 수 있으므로 오류의 파급을 최소로 줄일 수 있다. 특히, suffix로 사용되는 패턴 p가 적절하다면 오류는 두개의 부호어 이상 전파되지 않는다[8]. 따라서, ii)에 해당되는 부호화 방식은 오류제어기능을 갖는 정보원 부호화로 분류되고 그 대표적인 부호에 Fibonacci 부호[5][6]와 flag 부호[4]가 있다. 본 논문에서는 설명의 편의상 다음과 같은 기호를 사용한다.

B(n) 자연수 n의 2진표현이며, 최상위 비트(MSB)가 항상 1이다.

$\overleftrightarrow{B(n)}$ B(n)을 역순으로 재배열한 표현

0^m, 1^m m개의 0, m개의 1이 연속됨;
〈예〉 $1^2 0^3 = 11000$

$\bar{a}$ a의 보수. $\bar{0} = 1$, $\bar{1} = 0$

$B_x(n)$ x 방식에 의한 자연수 n의 2진 부호어

$L_x(n)$ $B_x(n)$의 평균부호길이

f flag 패턴의 길이 ($f \geq 2$)

이하 제Ⅱ장에서 평균부호길이면에서 우수한 것으로 알려져 있는 Fibonacci 부호에 대하여 살펴보고, 제Ⅲ장에서는 Wang과 Yamamoto가 제안한 두 가지 flag 부호에 대하여 고찰한다. 제Ⅳ장에서는 최적의 suffix 패턴을 갖는 flag 부호를 제안하여 그 성능을 Ⅱ장, Ⅲ장에서 설명한 부호들과 비트 삽입의 횟수 및 평균부호길이를 중심으로 비교·평가한다.

Ⅱ. Fibonacci 부호

Fibonacci 부호는 무한 자연수를 표현하는 방법의 하나로 부호화 과정에서 Fibonacci 표현을 사용한다. 이 부호는 정보의 내용을 어두(prefix)로 나타내고 부호어의 끝에 같은 패턴을 suffix로 부가하여 각 부호어 사이를 구별하도록 한다. 따라서, 부호어에 오류가 발생하여도 suffix 패턴에 의해 동기를 회복할 수 있다. 차수 m의 Fibonacci 표현은 식(6)과 같이 정의되는 Fibonacci 수(number)에 바탕을 둔다.

$$F_n = F_{n-1} + F_{n-2} + \cdots + F_{n-m},\ n \geq 1 \quad (6)$$
(단, $F_{-m+1} = F_{-m+2} = \cdots = F_{-2} = 0$, $F_{-1} = F_0 = 1$)

자연수 n의 Fibonacci 표현은

$$n = \sum_{n=1}^{k} d_i f_i,\ d_i \in \{0,1\}\ \ 0 \leq i \leq k \quad (7)$$

인 d_i의 계열로 나타낸다.

Apostolico와 Fraenkel이 제안한 Fibonacci 부호[5]는 부호화와 복호 과정에서 Fibonacci 수를 사용하고 01^m을 부호어 식별자인 suffix 패턴으로 사용하며, 가장 작은 자연수의 부호어로 1^m을 가진다(이하 AFF1 부호라 한다). AFF1 부호의 부호화 알고리즘은 아래와 같다.

〈AFF1's Encoding Algorithm〉

$n \in N^+ = \{1,2,\cdots\}$

(1) $n = 1$일 때 1^m,
$n = 2$일 때 01^m 할당

(2) $S_{k-2} < n \leq S_{k-1}$이 되는 k를 찾는다.
단, $S_k = \sum_{i=-1}^{k} F_i$ $(k \geq -1)$
여기서, $Q = n - S_{k-2} - 1$을 만족하는 Q를 Fibonacci 표현으로 나타낸다.

(3) suffix 패턴 01^m을 부가한다.

Fibonacci 부호 중 AFF1 부호보다 간단한 것에 가장 작은 자연수에 1^{m-1}을 할당하고 suffix로서 01^{m-1}을 사용하는 부호가 있다(이하 AFF2 부호라 한다). AFF2 부호는 AFF1 부호와는 달리 자연수 n 자체를 Fibonacci 표현으로 나타낸다. AFF2 부호의 가장 왼쪽은 항상 "1"이므로 실제로는 01^m의 패턴으로서 동기를 취하게 된다. 따라서, 부호어내에서 suffix 이외의 01^m이 나타나지 않도록 해야 한다.

AFF1 부호와 AFF2 부호는 부호어 내에 01^m을 허용하지 않는 일의(unique)의 suffix를 가지므로 오류에 대해 동기회복이 빠른 특징이 있다. 즉, 다음과 같이 m = 2로 부호화된 계열

$$\cdots 0011\ 1011\ 00011 \cdots$$

의 세번째 비트에서 오류가 발생하여

$$\cdots 00011\ 011\ 00011 \cdots$$

과 같이 복호되더라도 오류의 파급은 두 부호어내로 제한되게 된다(여기서, 비트열 사이의 공백은

부호어 간을 구별키 위해 편의상 삽입한 기호이다). 처음 두개의 부호어는 오류의 영향을 받더라도 세번째 부호어(00011)에서 파급이 중지됨을 알 수 있다. 그러나, AFF1, AFF2 부호의 최대 결점은 오류가 무한히 파급될 가능성이 있는 점이다[6]. 예를 들어

011 11 11 …11 1011

과 같이 부호화된 계열의 첫 비트에서 오류가 발생되었다면

11 11 11 11 … 11 011

로 복호되어 부호어가 끝날 때까지 오류가 전파되어 올바른 동기를 취할 수 없게 된다.

이러한 결점을 개선한 것이 Capocelli의 Fibonacci 부호이다[6](이하 CF 부호라 한다). CF 부호는 suffix로서 01^m을 사용하고, 01^m을 부호어의 어두에 허용하지 않는 어두 조건(prefix property)을 만족하는 부호로서 그 부호화 알고리즘은 다음과 같다.

〈CF's Encoding Algorithm〉

(1) n = 1일때 01^m 할당

(2) n ≠ 1일때

$$F_{k+5}-(k+5) < n \le F_{k+6}-(k+6)$$

을 만족하는 $k \ge 0$를 구한다.

(3) $F_{k+5}-(k+5) < n \le F_{k+5}-(k+5)+F_{k+2}$ 이면 길이가 k+4인 0α의 계열로 n을 부호화한다.

(단, α는 $n - F_{k+3} + 1$의 부호어)

(4) $F_{k+5}-(k+5)+F_{k+2} < n \le F_{k+6}-(k+6)$ 이면 길이가 k+4인 1β인 계열로 n을 부호화한다.

(단, β는 $n - F_{k+4} + 1$의 부호어)

$1 \le n \le 16$에 대한 AFF1, AFF2 및 CF 부호의 구성예를 표 1에 보인다.

표 1. Fibonacci 부호의 부호어(차수 m = 2)

n	AFF1	AFF2	CF
1	11	1	011
2	011	101	0011
3	0011	1001	1011
4	1011	10001	00011
5	00011	10101	01011
6	01011	100001	10011
7	10011	100101	11011
8	000011	101001	000011
9	001011	1000001	001011
10	010011	1000101	010011
11	100011	1001001	100011
12	101011	1010001	101011
13	0000011	1010101	110011
14	0001011	10000001	111011
15	0010011	10000101	0000011
16	0100011	10001001	0001011
avg.	5.3750	5.9378	5.4375

AFF1과 CF 부호의 평균부호길이 면에서의 성능은 두 부호의 밀도를 비교하므로써 알 수 있다. 밀도 D_k는 부호길이 l_k 이하의 부호어수라 두면, m = 2일때 AFF1 부호의 밀도 D_{k1}은

$$D_{k1} = \sum_{i=1}^{l_k-1} F_i = F_{l_k+1} - 1 \qquad (8)$$

이고, CF 부호의 밀도 D_{kc}는

$$D_{kc} = \sum_{i}^{l_k-1} (F_{i+2} - 1) = F_{l_k} - (l_k + 1) \qquad (9)$$

로 된다. 식(8)과 (9)의 결과 CF 부호의 밀도가 커서 동일 길이의 부호어로 더 많은 n을 표현할 수 있으므로 보다 효과적임을 알 수 있다.

이들 Fibonacci 부호는 점근적으로 최량은 아니지만 비교적 큰 자연수 n(n < 514,288)에 대해 평균부호길이면에서 다른 유니버셜 부호에 비해 우수한 것으로 알려져 있다[7].

Ⅲ. Flag 부호

자연수 n을 부호화하는 또 다른 방법에 부호어

의 끝을 나타내기 위하여 flag 패턴을 사용하는 부호가 있다[4]. Wang에 의해 제안된 flag 부호의 장점은 그 알고리즘이 매우 간단하면서 평균부호길이가 Fibonacci 부호에 근접됨을 들 수 있다. flag의 길이 f는 2이상이 되어야 하며, 자연수 n의 이진표현 B(n)은 $B(n) = b_0b_1 \cdots b_M$(이때 $b_0 = 1$)의 형태가 된다. Wang 부호의 부호화는 B(n)을 역으로 재배치한 $\overleftrightarrow{B(n)}$내에 (f - 1)개의 연속되는 0이 발생하면 1을 삽입(bit stuffing)하여 어두 조건을 만족시키며, 0'의 suffix를 부가하여 부호어간을 구별한다(이하, 이 부호를 $B_{W(f)}(n)$이라 한다).

〈예〉 n = 200, B(200) = 11001000, f = 3

(1) $\overleftrightarrow{B(200)}$ = 00010011

(2) $B_{W(3)}(200)$ = 00101001**11000**

(이때 1은 삽입된 비트, **000**는 flag 패턴)

$|B_{W(3)}(200)|$ = 13 [bit]

이와 같은 Wang의 알고리즘에 대해, 이진표현 B(n)을 역으로 취할때의 속도 저하를 방지하기 위해 Yamamoto 등은 B(n)의 역을 취하지 않는 개선방법 $B_{Y(f)}(n)$을 제안하였다[8]. 여기서 Yamamoto는 B(n)의 최상위 비트(MSB)가 아무런 정보도 갖고 있지 않다는데 주목하여 최상위 1비트를 제거하고 부호화함으로써 Wang 부호보다 평균부호길이를 단축하는 방법을 제시하였다. $B_{Y(f)}(n)$에서는 패턴 $p = p_1p_2 \cdots p_f$의 자기상관(autocorrelation)을 $pp = q_0q_1 \cdots q_{f-1}$이라 하고 식(10)과 같이 정의하여 suffix 선택의 기준으로 이용하였다.

$$q_j = \begin{cases} 1, & \text{if } p_s = p_{s+j}, \\ & 1 \le s \le f-j,\ 0 \le j < f \quad (10) \\ 0, & \text{others} \end{cases}$$

식(10)에 의해 발생되는 f = 3, 4에 대한 패턴 p와 자기상관값 pp의 관계를 표 2에 보인다.

표 2. 자기상관

f = 3		f = 4	
p	pp	p	pp
		0000	1111
		0001	1000
		0010	1001
		0011	1000
000	111	0100	1001
001	100	0101	1010
010	101	0110	1001
011	100	0111	1000
100	100	1000	1000
101	101	1001	1001
110	100	1010	1010
111	111	1011	1001
		1100	1000
		1101	1001
		1110	1000
		1111	1111

Yamamoto는 $pp = 10^{f-1}$를 만족하는 flag 패턴 $p = 10^{f-1}(f \ge 2)$, $p = 01^{f-1}(f \ge 2)$ 및 $p = 1^20^{f-1}(f \ge 4)$를 suffix로 사용하였다. Yamamoto 부호의 부호화는 B(n)과 flag 패턴 $p = p_1p_2 \cdots p_{f-1}$을 비교하여 일치하면 $\overline{p_1}$을 삽입하고, 최후미에 flag 패턴 $p = p_1p_2 \cdots p_f$을 연결한다. 부호화의 일례를 다음에 나타낸다.

〈예〉 n = 200, B(200) = 11001000,

f = 3, p = 100

$B_{Y(3)}(200)$ = 10**1**01010**1**00

$B_{Y(3)}(200)$ = 10**1**01010**1**00**100**

(이때 1은 삽입된 비트, **100**는 flag 패턴)

$|B_{Y(3)}(200)|$ = 12 [bit]

이 부호는 f = 2일때 $B_{W(f)}(n)$에 비해 평균부호길이가 평균 1[bit/codeword] 짧은 것으로 알려져 있으나[8], 자기상관을 이용한 flag 패턴에 의해서라기 보다는 MSB 삭제에 의한 효과로 평가된다.

Ⅳ. 제안 flag 부호 및 성능 평가

Ⅲ장에서 살펴 본 flag 부호의 경우, 평균 부호길이는 비트 삽입의 발생횟수에 많은 영향을 받음을 알 수 있다. 따라서, 본 논문에서는 비트 삽입이 최소로 발생하는 flag 패턴을 구하고 그 성능을 기존의 부호들과 비교·분석한다.

자기상관값이 큰 패턴일수록 평균부호길이가 더 짧다는 사실[18]을 고려할 때 Yamamoto 부호에서 패턴 p의 선택조건을 만족하는 패턴은 $0^{f-1}1$과 $1^{f-1}0$임을 알 수 있다. 일반적으로 $0^{f-1}1$보다 $1^{f-1}0$의 패턴이 자연수의 2진표현 내에 나타나는 횟수가 적기 때문에 $1^{f-1}0$의 패턴을 선택하여 Yamamoto 부호에 적용한 부호를 수정된 Yamamoto 부호; $B_{MY(f)}(n)$이라 한다. 그 부호화 알고리즘을 아래에 나타낸다.

〈Modified Yamamoto's Encoding Algorithm〉

(1) 초기화
  ① $B(n) = b_0b_1 \cdots b_M$,
    $M = \lfloor \log_2 n \rfloor$,
    $\lfloor x \rfloor$: x보다 크지 않은 최대정수
  ② $B_{MY(f)}(n) \leftarrow \Lambda$
  ③ $t = 1$ (b_0 통과)
(2) 삽입
  ① if $t > M - f + 2$
    then [if $t \le M$,
      then $B_{MY(f)}(n) \leftarrow b_tb_{t+1}\cdots b_M$]
    goto (3)
  ② if $b_tb_{t+1}\cdots b_{t+(f-2)} \neq p_1p_2\cdots p_{f-1}$
    then $B_{MY(f)}(n) \leftarrow b_t$, $t = t + 1$
    else $B_{MY(f)}(n) \leftarrow b_tb_{t+1}\cdots b_{t+(f-2)}\overline{p_f}$,
      $t = t + (f - 1)$
  ③ goto (2)
(3) flag 패턴 $p_1p_2\cdots p_f$ 연결
  $B_{MY(f)}(n) \leftarrow p_1p_2\cdots p_f$

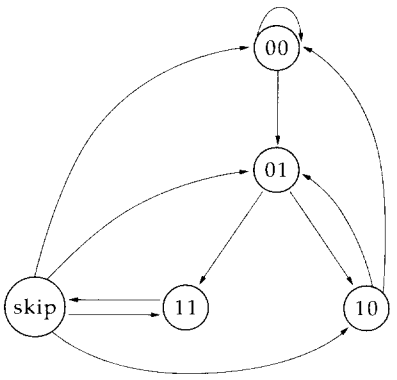

그림 1. 제안 부호의 상태천이도

❖ 정리 1

$B(n) = b_0b_1 \ldots b_M$일때, 각 $b_j(1 \le j \le M)$가 $Pr\{b_j = 0\} = Pr\{b_j = 1\} = 1/2$의 확률로 독립적으로 발생하면 $B_{MY(f)}(n)$의 비트 삽입 발생횟수 $C(B_{MY(f)}(n))$은

$$C(B_{MY(f)}(n)) = \begin{cases} 0, & M \le f-2 \\ \dfrac{M-f+2}{2^f}, & M \ge f-1 \end{cases} \quad (11)$$

이다.

증 명

예를 들어 $f = 3$일때 $p = 110$로 된다. $M \le f - 2$이면 비트 삽입이 발생하지 않고, $M > f - 2$이면 $j = 1$에 대하여 $Pr\{b_jb_{j+1} = 00\} = Pr\{b_jb_{j+1} = 01\} = Pr\{b_jb_{j+1} = 10\} = Pr\{b_jb_{j+1} = 11\} = 1/4 = 1/2^{f-1}$이다. $b_jb_{j+1} = 11$이면 $\overline{p_f}(=1)$이 삽입되고, 다시 $b_{j+2}b_{j+3}$이 체크된다. $b_jb_{j+1} \neq 11$이면 $b_{j+1}b_{j+2}$가 체크된다. 따라서, $b_jb_{j+1}(1 \le j \le M - f + 2)$의 상태천이도는 그림 1과 같이 된다. 그림 1로부터 $1 \le j \le M - f + 2$에 대한 각 비트열의 발생확률은 식(12)와 같다.

$$\begin{aligned} Pr\{00\} &= Pr\{01\} = 1/4 = 1/2^{f-1} \\ Pr\{10\} &= Pr\{11\} = 1/8 = 1/2^f \end{aligned} \quad (12)$$

부호어내의 2진열과 flag 패턴은 최대 (M-f+2)회 비교되므로 비트 삽입이 발생할 확률 $1/2^f$를 곱하면 비트 삽입의 발생횟수는 $\frac{1}{2^f}(M-f-2)$로 된다. [증명끝]

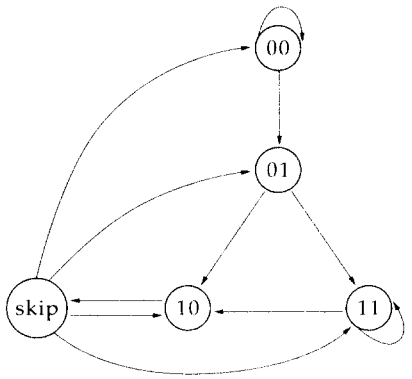

그림 2. Yamamoto 부호의 상태천이도

❖ 정리 2

정리 1과 같은 조건하에서 $B_{Y(f)}(n)$의 비트 삽입 발생횟수 $C(B_{Y(f)}(n))$은

$$C(B_{Y(f)}(n)) = \begin{cases} 0, & M \le f-2 \\ \frac{M-f+2}{2^{f+1}}, & M \ge f-1 \end{cases} \quad (13)$$

이다.

☞ 증 명

$f = 3$일때 $p = 110$로 된다. 이때 $b_j b_{j+1}(1 \le j \le M-f+2)$의 상태천이도인 그림 2로부터

$$\begin{aligned} Pr\{00\} &= Pr\{01\} = 1/8 = 1/2^f \\ Pr\{10\} &= Pr\{11\} = 1/4 = 1/2^{f-1} \end{aligned} \quad (14)$$

임을 알 수 있다. 비트 삽입 발생횟수는 $\frac{1}{2^{f-1}}(M-f+2)$로 된다. [증명끝]

❖ 정리 3

$f = 3$일때 $p = 000$가 되는 $B_{W(f)}(n)$의 비트 삽입 발생횟수 $C(B_{W(f)}(n))$는

$$C(B_{W(f)}(n)) = \begin{cases} 0, & M \le f-2 \\ \frac{M-f+2}{2^f}, & M \ge f-1 \end{cases} \quad (15)$$

이다.

☞ 증 명

정리 1, 정리 2와 같은 방법으로 증명된다.

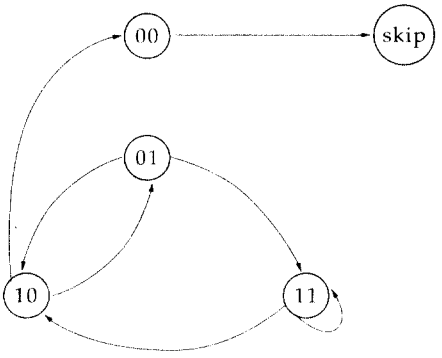

그림 3. Wang 부호의 상태천이도

또한, $f \ge 4$에 대해서도 각 부호의 비트 삽입 횟수는 정리 1, 2, 3으로 된다.

위의 정리 1, 정리 2, 정리 3으로부터 비트 삽입의 횟수는 점근적으로 볼 때

$$C(B_{MY(f)}(n)) = C(B_{W(f)}(n)) < C(B_{Y(f)}(n))$$

순으로 일어나게 되나, 유한의 n에 대해서는 $C(B_{MY(f)}(n))$이 더 적게 일어남을 표 3의 실험 결과로 알 수 있다. 한편, $B_{W(f)}(n)$는 $B_{Y(f)}(n)$보다 비트 삽입이 적게 일어나지만 최상위 1비트를 제거하지 않으므로 평균부호길이면에서는 $B_{Y(f)}(n)$보다 평균부호길이가 조금 더 길게 나타난다. 이와 같은 관계로부터 각 부호 방식의 평균부호길이는 다음과 같이 나타낼 수 있다.

$$L_{MY(f)}(n) < L_{Y(f)}(n) < L_{W(f)}(n)$$

그러나, 실제로 자연수의 2진표현 B(n)내에 0과 1이 정확히 1/2의 확률로 분포하고 있지 않으므로 0^f의 flag 패턴보다는 $1^{f-1}0$의 제안 패턴을 사용하는 것이 비트 삽입이 적게 발생한다. $1 \leq n \leq 10^5$인 자연수 n에 대한 비트 삽입 발생횟수를 표 3에 보인다.

적응 사전 부호화 방식의 실제 사용범위인 유한의 $n \leq 10^5$에 대하여 CF 부호와 제안 B_{MY} 부호의 평균부호길이를 표 4에 나타내었다. 표 4의 결과, 제안한 B_{MY} 부호가 CF 부호보다 월등히 간단한 알고리즘으로 CF 부호와 대등한 평균부호길이를 가짐을 알 수 있다. 특히, $n > 10^3$에 대해 제안 부호는 CF 부호보다 평균부호길이가 짧아지는 효과를 얻게 된다.

표 5는 f = 3일때, $1 \leq n \leq 256$인 자연수 n에 대한 각 부호방식의 부호어의 일부를 나타낸 것으로 평균부호길이를 avg.의 [bits/codeword]로 비교하였다.

V. 결 론

본 논문에서는 정보원 부호화 단계에서 오류를 제어할 수 있는 자기 동기 부호에 대하여 Fibonacci 부호와 flag 부호를 중심으로 살펴보았다. flag 부호에 있어서 Yamamoto 부호에서 사용할 수 있는 flag 패턴 중 $1^{f-1}0$의 최적의 flag 패턴을 구하여 개선된 flag 부호를 구성하였으며, 그 성능을 비트 삽입 발생횟수와 평균부호길이면에서 분석하였다.

제안부호는 자연수의 2진표현내에서 비트 삽입이 가장 적게 발생하고, 이를 상태천이도를 통하여 분석한 결과 Yamamoto 부호에 비해 비트 삽입 발생확률이 1/2로 줄어듦을 알 수 있다. 또한, 비트 삽입 횟수가 감소하므로써 평균부호길이면에서도 그 성능이 Yamamoto 부호에 비해 향상되었다. flag 부호 방식 중 알고리즘의 복잡도와 평균부호길이면에서 가장 우수한 결과를 보이는 제안부호와 평균부호길이면에서 성능이 가장 우수한 것으로 알려져 있는 CF 부호의 평균부호길이를 비교해 보면 f가 3, 4이고 $n \geq 10^3$일때 제안 부호의 성능이 CF 부호보다 우수하게 나타났다.

본 논문에서 고찰한 부호의 성능은 n의 발생확률이 내림차순 분포일때만을 고려하였으나 다양한 정보원 확률분포를 고려한 분석과 최소 조건을 만족시키는 부호의 구성법이 향후의 과제로 남아 있다.

표 3. $pp = 10^{f-1}$이 되는 각 패턴에 대한 비트 삽입의 횟수

f	패턴 p	최대수 n				비 고
		10^2	10^3	10^4	10^5	
3	001	78	1,241	19,176	241,368	B_W
	011	101	1,770	25,959	349,172	
	100	88	1,759	25,055	333,637	B_Y
	110	60	1,186	17,008	226,538	B_{MY}
4	0001	29	478	7,782	98,537	B_W
	0011	41	764	12,643	162,998	
	0111	38	760	11,610	161,894	
	1000	35	759	11,287	155,231	
	1100	29	743	11,158	153,278	B_Y
	1110	20	446	6,645	91,577	B_{MY}

표 4. 평균부호길이 비교

[bits/codeword]

f	부호방식	최대수 n			
		10	10^2	10^3	10^4
3	B_{MY}	5.000	8.400	12.173	16.064
	CF	4.900	8.240	12.538	17.248
4	B_{MY}	5.900	9.000	12.433	16.028
	CF	5.900	8.860	12.478	16.213

표 5. 각 부호 방식의 부호어 (f = 3)

n	$CF_{(3)}(n)$	$B_{W(3)}(n)$	$B_{Y(3)}(n)$	$B_{MY(3)}(n)$
1	011	1000	100	110
2	0011	01000	0100	0110
3	1011	11000	1100	1110
4	00011	001_1000	00100	00110
5	01011	101000	01100	01110
6	10011	011000	101100	10110
7	11011	111000	11100	111_110
8	000011	00101000	000100	000110
9	001011	10011000	001100	001110
10	010011	0101000	0101100	010110
50	000100011	0100111000	1010101100	10010110
100	0101010011	001100111000	10101010100	100100110
150	00100100011	011010011000	001011101100	00101110110
200	11010100011	0010100111000	101010100100	1001000110
256	010000000011	0010010010011000	000000000100	000000000110
avg.	9.929688	10.945312	10.292969	9.929688

※ 단, "_"은 삽입된 비트

참 고 문 헌

[1] T.C.Bell, J.G.Cleary, and I.H. Whitten, "Text compression," pp. 318 *Prentice Hall*, (1990)

[2] Y.Miyakawa, J.H.Park, and H.Imai, "Universal code with self-synchronization," IEICE Spring conf., SA-6-1 (Mar. 1990) (in Japanese)

[3] P.Elias, "Universal codeword sets and the representation of the integers," *IEEE Trans. Inform. Theory*, vol.IT-21, No.2, pp. 194-203 (Mar. 1975)

[4] M. Wang, "Almost asymptotically optimal flag encoding of the integers," *IEEE Trans. Inform. Theory*, vol.IT-34, No.2, pp. 324-326, (Mar. 1988)

[5] A.Apostolico, A.S.Fraenkel, "Robust transmission of unbounded strings using Fibonacci representations," *IEEE Trans. Inform. Theory*, vol.IT-34, No.2, pp. 238-245 (Mar. 1987)

[6] R.M.Capocelli, "Comments and additions to 'Robust transmission of unbounded strings using Fibonacci representations'," *IEEE Trans. Inform. Theory*, vol.IT-35, No.1, pp 191-193 (Jan. 1989)

[7] D.A.Lelwer, D.S.Hirschberg, "Data compression," *ACM Computing Surveys*, vol.19, No.3, pp. 274-276 (Sep. 1987)

[8] H.Yamamoto, H.Ochi, "A new asymptotically optimal code for the positive integers," *IEEE Trans. Inform. Theory*, vol.IT-37, No.5, pp. 1420-1429 (Sep. 1991)

[9] R.M.Capocelli, A.De Santis, "Regular universal codeword sets," *IEEE Trans. Inform. Theory*, vol.IT-32, No. 1, pp. 129-133 (Jan. 1986)

□ 著者紹介

이 선 영 (李 善 英, Lee Sun Young) 학생회원

1993년 2월 부산수산대학교 전자계산학과 (이학사)
1993년 - 현재 부산수산대학교 대학원 전자계산학과 석사과정

※ 주관심분야 : 정보이론, 데이타압축, 컴퓨터그래픽스, 영상처리 등

박 지 환 (朴 志 煥, Park Ji Hwan) 정 회 원

1984년 2월 경희대학교 전자공학과 (공학사)
1987년 3월 일본 국립전기통신대학 대학원 정보공학과 (공학석사)
1990년 3월 일본 요코하마국립대학 대학원 전자정보공학과 (공학박사)
1990년 - 1992년 부산수산대학교 전자계산학과 전임강사
1992년 - 현재 부산수산대학교 전자계산학과 조교수

※ 주관심분야 : 정보이론, 데이타압축, 컴퓨터그래픽스, 영상처리 등