通信情報保護學會誌
第3卷 第1號, 1993. 3

랜덤 치환 고속 발생기 설계 및 응용

A Pseudo Random Permutation Generator with application to random bit generator

고승철* · 이대기*

요 약

본 논문에서는 Akl-Meijer가 설계한 랜덤 치환 발생기를 일반화한 알고리즘을 제안한다. Akl과 Meijer는 사이즈가 m인 치환(Permutation)과 0과 $m!$-1 사이의 정수를 일대일 대응시키는 Knuth의 알고리즘을 이용하여, 선형 합동법 $Y=X+C \bmod m!$ (C는 상수)에서 발생되는 난수와 일대일 대응되는 치환을 고속으로 계산하는 기법을 제시하고, 이 기법에 의해 발생된 치환을 n개를 결합하여 랜덤 치환을 발생하는 치환 발생 알고리즘을 설계하였으며, 이를 응용하여, 이진 난수 발생기를 제시하였다. 본 논문에서는 선형 합동법 $Y=AX+C \bmod m!$ (A, C는 상수)에서 발생되는 난수와 일대일 대응되는 치환 계산과정을 상삼각 행렬(Upper triangular matrix)의 곱으로 변환하여 고속으로 계산하는 알고리즘을 제시한후, 이 알고리즘의 출력 치환을 n개 결합하여 치환을 발생하는 랜덤 치환 발생기를 설계한다. 또한 이의 암호적인 응용으로, 치환 발생기를 이용한 이진 난수 발생기를 제시한다.

1. 서 론

난수열 즉 랜덤하게 선택된 수의 수열은 과학적 공학적인 분야에서 응용되어 왔다. 예를들면, 핵물리 또는 대기체계(Queueing) 이론에서의 컴퓨터 모의검증, 샘플링 이론, 수치 해석에서 발생되는 제반 문제의 해법, 컴퓨터 S/W의 우수성 판정등이 주요 응용 분야이다[2].

난수 특성은 암호화에 사용되는 수열이 갖추어야 할 가장 기본적인 특성 중 하나이다. 현대 암호 시스템은 통신 및 컴퓨터 보안을 주 응용 대상으로 발전되었기 때문에, 이진 난수열(Binary random sequence)를 주요 연구 대상으로 고려하여 왔다. 그러나, 일반적으로 난수란 십진 난수가 보편적이며, 암호 기술 측면에서도 십진 난수는 여러 방면에서 응용이 가능하다.

이진 난수 발생기로서는 주로 원시 다항식(Primitive polynomial in GF(2^n))에 근거한 Linear

* 한국전자 통신연구소

Feedback Shift Register(LFSR)을 이용하여 왔으며[1], 일반적인 난수 발생기로서는 선형 합동법(Linear Congruential Method)을 주로 이용하여 왔다[2]. 그러나 이런 방법들은 난수의 통계적 특성은 우수하지만 선형적인 성질 때문에 암호 수열로서는 사용이 불가능하다.

암호 기술에서 응용 가능한 난수 발생기는 먼저 암호적으로 안전(Cryptographically secure) 하여야 하며, 고속으로 난수 발생이 가능하여야 한다. 암호적으로 안전한 수열이란, 부분 수열의 정보를 이용하여 그 수열의 다른 부분을 예측하는 것이 계산상 불가능한 수열을 의미한다[3].

Akl-Meijer는 선형 합동법이 제공하는 주기성 및 난수의 통계적 특성을 보존하는 랜덤 치환 발생기를 제안하였다[4]. 사이즈가 m 임의의 치환과 0과 $m!$-1 사이의 정수가 일대일 대응된다는 성질을 이용하여, 선형 합동법에 의해 발생되는 난수에 대응되는 치환들을 결합하여 랜덤 치환을 발생하는 것이 그들이 제안한 발생기의 핵심 원리이다. 그들은 S/W 구현이 용이하고 고속 랜덤 치환이 가능하도록 $Y=X+C \bmod m!$ 를 난수 발생 수식으로 사용하였다.

본 논문에서는 선형 합동법 $Y=AX+C \bmod m!$ (A, C는 상수)에서 발생되는 난수와 일대일 대응되는 치환 계산과정을, 벡터와 상삼각 행렬(Upper triangular matrix)의 곱으로 변환하여 고속으로 계산하는 알고리즘을 제시한 후, 이 알고리즘의 출력 치환을 n개 결합하여 치환을 발생하는 랜덤 치환 발생기를 설계한다. 치환 계산 과정에서 소요되는 계산량은 곱셈과 덧셈이 각각 $\frac{(m-1)m}{2}$ 이다. Akl-Meijer 기법과 마찬가지로 우리가 제안한 기법도 다정도 연산이 필요없고, 랜덤 치환을 고속으로 발생할 수 있으며, S/W 구현이 매우 용이하면서도 Akl-Meijer기법을 일반화하였으므로 비도가 상대적으로 우수하며 난수의 통계적 특성 역시 우수하다.

2장에서는 치환 발생기의 기본 원리인 계승 수 체계(Factorial Number System, FNS)를 설명하며, 3장에서는 선형 합동법 $Y=AX+C \bmod m!$ (A, C는 상수)에서 발생되는 난수와 일대일 대응되는 치환을 계산하는 기법을 제시한다. 4장에서는, 랜덤 치환을 고속으로 발생하는 구체적인 알고리즘을 제시한다. 우리가 제안한 알고리즘들의 암호적인 평가 및 분석은 현재 진행중이다.

2. 계승 수 체계와 치환

2.1. 계승수체계

일반적으로 수는 십진법으로 표현되지만, 컴퓨터 및 현대 통신 시스템의 발달로 이진법으로 수를 표현하는 것도 보편화 되었다. 이와같은 표현법들이 자릿수마다 동일 진법을 사용하는데 비하여, 수를 자릿수마다 서로 다른 진법으로 표현할 수도 있다. 이런 표현 방법을 혼합 진법(Mixed Radix System)이라고 한다[2]. 혼합 진법중 대표적인 예가 바로 계승 수 체계(Factorial Number System, FNS)에 의한 수의 표현법이다.

주어진 m(m은 양정수)에 대하여, $n(0 \le n < m!)$을 다음과 같이 계승 수 체계에 의해 표현한다.

$$\begin{aligned} n &= a_1 + 2\times(a_2 + 3\times(a_3 + \cdots + (m-2)\times \\ &\quad (a_{m-2} + (m-1)\times a_{m-1})\cdots) \\ &= a_1 1! + a_2 2! + a_3 3! + \cdots + a_{m-2}(m-2)! + \\ &\quad a_{m-1}(m-1)!,\ 0 \le a_i \le i \end{aligned}$$

예를들면, m=5인 경우 n=100은 다음과 같이 표현된다.

$$\begin{aligned} 100 &= 0 + 2\times(2 + 3\times(0 + 4\times 4)) \\ &= 0\times 1! + 2\times 2! + 0\times 3! + 4\times 4! \end{aligned}$$

Algorithm : 주어진 m(m은 양정수)에 대하여, $n(0 \le n < m!)$의 계승 수 체계에 의한 표현법

For i=2, 3, ···, m

Set $a_{i-1} \leftarrow n \bmod i$ and $n \leftarrow \lfloor n/i \rfloor$

2.2. 치환 발생 알고리즘

계승 수 체계에 의해 표현된 수와 치환(Permutation)은 일대일 대응관계가 있다. 수와 치환을 일대일 대응시키는 Knuth의 알고리즘을 소개한다[2].

Algorithm : 초기 치환 P=(u_1, u_2, ⋯, u_m)주어졌을 때,

$n=a_1 1!+a_2 2!+\cdots+a_{m-1}(m-1)!$, $0\le a_i\le i$에 대하여,

For $r=1, 2, \cdots, m-1$

Exchange $u_{r+1} \longleftrightarrow u_{a_r+1}$.

일반적으로 초기 치환으로 I=(1, 2, ⋯, m)을 선택하는 것이 보편적이다. 알고리즘의 동작과정을 예를들어 설명한다.

$n=100=0\times1!+2\times2!+0\times3!+4\times4!$에 대응되는 치환은

초기 치환 P_0=I=(1, 2, 3, 4, 5)

$a_1=0 \rightarrow P_1=(2, 1, 3, 4, 5)$

$a_2=2 \rightarrow P_2=(2, 1, 3, 4, 5)$

$a_3=0 \rightarrow P_3=(4, 1, 3, 2, 5)$

$a_4=4 \rightarrow P_4=(4, 1, 3, 2, 5)$

3. 랜덤 치환 계산법

선형 합동법 $Y=AX+C \bmod m!$ (A, C : 상수)에서 발생되는 난수와 일대일 대응되는 치환을 고속으로 계산하는 기법을 제시한다.

이제 X와 C를 계승 수 체계상에서 다음과 같이 표현된다고 하자.

$$X=\sum_{i=1}^{m-1} x_i i! = x_1 1!+x_2 2!+\cdots+x_{m-1}(m-1)!, \quad 0\le x_i\le i,\ i=1, \cdots, m-1$$

$$C=\sum_{i=1}^{m-1} c_i i! = c_1 1!+c_2 2!+\cdots+c_{m-1}(m-1)!, \quad 0\le c_i\le i,\ i=1, \cdots, m-1$$

이 때,

$$Y=AX+C=A\left(\sum_{i=1}^{m-1} x_i i!\right)+\left(\sum_{i=1}^{m-1} c_i i!\right)$$

$$=\left(\sum_{i=1}^{m-1} x_i Ai!\right)+\left(\sum_{i=1}^{m-1} c_i i!\right) \bmod m! \qquad (1)$$

수식 (1)에서 A가 상수이면, $Ai!$ ($i=1, \cdots, m-1$) 역시 고정된 상수라는 점이 우리가 제안하는 기법의 Key idea이다. 이제 $i=1, \cdots, m-1$에 대하여, $Ai!$이 다음과 같이 계승 수 체계에 의해 표현된다고 하자.

$$Ai! = a_{i1}1!+a_{i2}2!+\cdots+a_{im-1}(m-1)!, \quad 0\le a_{ij}\le j$$

계승 수 체계 표현법에 의해, 모든 i에 대하여 $a_{ij}=0$, $1\le j<i$. 그러므로 고정된 상수 A에 대하여, 상 삼각 행렬(Upper triangular matrix) H를 다음과 같이 유일하게 결정할 수 있다.

$$H=\begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots\cdots & a_{1m-2} & a_{1m-1} \\ 0 & a_{22} & a_{23} & \cdots\cdots & a_{2m-2} & a_{2m-1} \\ & 0 & a_{33} & \cdots\cdots & & \\ & & 0 & \cdots\cdots & & \\ & & & \cdots\cdots & & \\ & & & \cdots\cdots & & \\ 0 & & & \cdots\cdots & 0 & a_{m-1m-1} \end{pmatrix} \qquad (2)$$

또한,

$$\begin{aligned} Y &= AX+C \bmod m! \\ &= \sum_{i=1}^{m-1} x_i\left(\sum_{j=1}^{m-1} a_{ij} j!\right)+\sum_{j=1}^{m-1} c_j j! \bmod m! \\ &= \sum_{j=1}^{m-1}\left(\sum_{i=1}^{m-1} x_i a_{ij} j!\right)+\sum_{j=1}^{m-1} c_j j! \bmod m! \\ &= \sum_{j=1}^{m-1}\left(\sum_{i=1}^{m-1} x_i a_{ij}+c_j\right) j! \bmod m! \end{aligned} \qquad (3)$$

이제 벡터 $\vec{X}$, $\vec{C}$, $\vec{Y}$를 각각

$\vec{X}=(x_1, x_2, \cdots, x_{m-1})$, $\vec{C}=(c_1, c_2, \cdots, c_{m-1})$,
$\vec{Y}=(y_1, y_2, \cdots, y_{m-1})$

이라고 정의하면, 수식 (3)으로 부터, 선형 합동법 $Y=AX+C \bmod m!$을 다음과 같이 행렬과 벡터의 곱으로 표현할 수 있다.

$$\vec{Y}=\vec{X}H+\vec{C} \qquad (4)$$

수식 (4)의 계산에 소요되는 연산량은 곱셈과 덧셈이 각각 $\frac{(m-1)m}{2}$ 이다.

4. 랜덤 치환 발생기 설계

계승 수 체계에 의해 표현된 수와 치환을 일대일 대응시킬 수 있다는 성질을 이용하여, Akl-Meijer는 선형 합동법 $Y=AX+C \bmod m!$ 에서 보장하는 주기성과 난수 특성을 보존하는 랜덤 치환 발생기를 제안하였다.

이제 3장에서 설명한 기법을 사용하여 고속으로 랜덤 치환들을 발생하고, 그 치환들을 결합하여 랜덤 치환을 발생하는 알고리즘을 제시한다. 실질적인 치환 발생 과정은, 먼저 알고리즘 PERGEN_DRIVE를 구동시켜, 알고리즘 PERGEN 구동의 준비를 마친후, 알고리즘 NONPERGEN을 구동시킨다.

Algorithm PERGEN_DRIVE(치환 발생기 구동의 준비 단계)

Input

$Y=AX+C \bmod m!$: 치환 발생기의 난수 발생용 선형 합동식

Output

$H=(a_{ij})$: a $(m-1)\times(m-1)$ matrix,
$0\le i, j \le m-2$, $0\le a_{ij} \le j+1$, $a_{ij}=0$ if $j<i$.
$\vec{C}=(c_i)$: a vector, $0\le i \le m-2$, $0\le c_i \le i+1$.

P1. Set $j \leftarrow 0$.
P2. Set $c_j \leftarrow C \bmod j+2$ and $C \leftarrow \lfloor C/(j+2) \rfloor$.
P3. Increase j by one. Now if $j<m-1$ go back to tep P2.
P4. Set $i \leftarrow 0$.
P5. Set $j \leftarrow i$ and *atemp* $\leftarrow$ A.
P6. Set $a_{ij} \leftarrow i$ *atemp* $\bmod j+2$ and *atemp* $\leftarrow \lfloor atemp/(j+2) \rfloor$.
P7. Increase j by one. Now if $j < m-1$ go back to step P6.
P8. Increase i by one. Now if $i < m-1$ go back to step P5.

Algorithm PERGEN(치환 발생기)

Input

m : a non-negative integer.
$H=(a_{ij})$: a $(m-1)\times(m-1)$ matrix,
$0\le i, j \le m-2$, $0\le a_{ij} \le j+1$, $a_{ij}=0$ if $j<i$.
$X=(x_i)$: a vector, $0\le i \le m-2$, $0\le x_i \le i+1$.
$C=(c_i)$: a vector, $0\le i \le m-2$, $0\le c_i \le i+1$.

Output

A permutation P uniquely determined by the integer $Y=AX+C \bmod m!$, where A, X and C are intgers determined uniquely by the matrix H, and vectors X, C respectively. X will be replaced by Y.

P1. [Initialize] Set carry $\leftarrow 0$. For $j=0, \cdots, m-2$, $y_j \leftarrow c_j$. And for $j=0, \cdots, m-1$, $u_j \leftarrow j$.
P2. [Initialize j] Set $j \leftarrow 0$.
P3. [Add carry and reset] Set $y_j \leftarrow y_j+carry$, then $carry \leftarrow 0$.
P4. [Initialize i] Set $i \leftarrow 0$.
P5. [Zero multiplier ?] IF $x_i=0$, go to step P7.
P6. [Multiply and add] Set $y_j \leftarrow y_j+x_i a_{ij}$, then set $carry \leftarrow carry+y_j/(j+2)$ and $y_j \leftarrow y_j \bmod j+2$.
P7. [Loop on i] Increase i by one. Now if $i \le j$, go back to step P5.
P8. [Exchange] Exchange $u_{j+1} \Longleftrightarrow u_{y_j}$.
P9. [Loop on j] Increase j by one. Now if $j < m-1$ go back to step P3.
P10. [Replace X by Y] For $j=0, \cdots, m-2$, $x_j \leftarrow y_j$.

Algorithm NONPERGEN(랜덤 치환 발생기)

Call PERGEN k times to generate permutations $P_1, \cdots, P_k$.
Output : $P=P_1 \circ P_2 \circ \cdots \circ P_k$.

우리는 위의 알고리즘을 선형 합동법 $Y=AX+C \bmod 100!$ 을 사용하여 SUN4-SPARC 기종에서 C 언어로 구현하였다. A와 C는 선형 합동법의 조건을

만족하는 수를 랜덤하게 선택하였다.

Algorithm : Random bit generator

Input
  $n \leftarrow 100$
  $k \leftarrow 2$
  $b_0, b_1, \cdots, b_{n-1} \leftarrow 0, 1, 0, 1, \cdots, 0.1$
repeat as many times as required
  Call NONPERGEN to generate P.
  for $i = 0, 1, \cdots, n-1$
    exchange $(b_i, b_{P(i)})$
  Output : $b_0, b_1, \cdots, b_{n-1}$
end repeat

5. 결 론

Akl-Meijer의 랜덤 치환 발생 기법을 일반화하여, 랜덤 치환을 발생하는 알고리즘 NONPERGEN을 설계하였다. 알고리즘 NONPERGEN은 선형 합동법 $Y=AX+C \mod m!$ (A, C : 상수)에서 발생되는 난수와 일대일 대응되는 치환 계산과정을, 벡터와 상삼각 행렬(Upper triangular matrix)의 곱으로 변환하여 고속으로 계산한 후, 그 출력 치환을 n개 결합하여 치환을 발생하는 랜덤 치환 발생기이다. 치환 계산 과정에 소요되는 계산량은 곱셈과 덧셈이 각각 $\frac{(m-1)m}{2}$ 이다.

Akl-Meijer 기법과 마찬가지로 우리가 제안한 기법도 다정도 연산이 필요없고, 랜덤 치환을 고속으로 발생할 수 있으며, S/W 구현이 매우 용이하면서도 Akl-Meijer 기법을 일반화하였으므로 비도가 상대적으로 우수하며 난수의 통계적 특성 역시 우수하다. 단, 연속적인 $n-1$개의 비트를 이용하여, n번째 비트를 완전히 예측할 수 있다는 암호적인 취약점이 있다. 우리가 제안한 알고리즘의 암호적인 평가 및 분석은 현재 진행중이다.

참 고 문 헌

1. W.W. Peterson and E.J. Weldon, Jr., ERROR-CORRECTING CODES, The MIT Press, 1972.

2. D.E. Knuth, The Art of Computer Programming, Vol. 2 : Seminumerical Algorithms, Addison-Wesley, 1981.

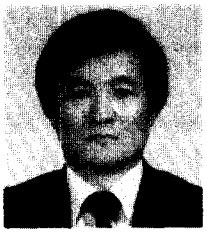
3. A. Shamir, "On the generation of cryptographically strong pseudorandom sequences, ACM Tran. on Computer Systems, Vol. 1, No. 1, 1983.

4. Selim G. Akl and Henk Meijer, "A Fast Pseudo Random Permutation Generator With Applications to Cryptology", Advances in Cryptology : Proceedings of CRYPTO 84, pp.269-275, 1985.

□ 著者紹介

고 승 철(정회원)

1981년 연세대학교 수학과(학사)
1983년 연세대학원 수학과(석사)
1992년 포항공대 수학과(박사)
1984년~현 재 : ETRI 선임연구원



이 대 기(정회원)

1966년 한양대학교 전기공학과(학사)
1987년 한양대학교 전자공학과(석사)
1980년~1992년 ETRI 산업기술개발부장, 지상시스템연구부장
1992년~현 재 : ETRI 부호기술부장
한국통신정보보호학회 산학이사