The 11th Workshop on Finite Field Theory and its Application for Pseudo-Random Sequences (FFTPRS2025)
Hita (Hybrid), September 16-17, 2025

安定した同期のための 4 次元 Lorenz-Stenflo 方程式の整数変換手法
A Study on a Pseudo-Random Bit Sequence Sharing Method Using an Integerized 4-Dimensional Lorenz-Stenflo Equation

髙市 康平 *† 顔 綿柱 ‡ 宮﨑武 § 上原 聡 ¶ 荒木 俊輔 *
Kohei Takaichi Jun-Juh Yan Takeru Miyazaki Satoshi Uehara Shunsuke Araki

Abstract— This study addresses the issue of computational order dependency in a dynamic random key generator. This system facilitates secure communication by generating numerous keys for a sender and receiver through the convergence of their respective 4-dimensional Lorenz-Stenflo equation values. To mitigate this dependency, we introduced an integer transformation. We show that synchronization between the two parties is maintained post-transformation and verify that the outcome is independent of the order of calculations.

Keywords— 4-dimensional Lorenz-Stenflo equation, Synchronized dynamic key generators, Integer transformation

1 はじめに

近年, リモートワークの普及などによって通信技術への需要が高まっている. そんな中 Lin らは動的ランダム鍵生成器 [1] を提案した. これはローレンツ方程式を拡張した式である 4 次元 Lorenz-Stenflo 方程式を用いて送信者と受信者で大量の鍵を生成し, 共通鍵暗号化方式を用いて秘密通信を行うシステムである.

このとき, 先行研究では浮動小数点演算を用いて 4 次元 Lorenz-Stenflo 方程式の計算を行っている. しかし浮動小数点演算を用いる従来手法では, 同期信号の処理方法やコンピュータ上での計算順序といった実装方法によっては数式的に等価なはずの計算結果が変動する. この違いによって送信側と受信側で異なる鍵が生成され, 通信を失敗させる原因となる.

上記の問題を解消するため, 本研究では 4 次元 Lorenz-Stenflo 方程式を完全に整数変換することを提案する. そして, 提案手法の計算結果が内部の実装方法に依存しないことを実証する.

* 九州工業大学, 福岡県飯塚市川津 680-4, Kyushu Institute of Technology, 680-4 Kawazu, Iizuka, Fukuoka, Japan.
† takaichi.kohei283@mail.kyutech.jp
‡ 国立勤益科技大学, 臺中市太平區坪林里中山路二段 57 號, National Chin-Yi University of Technology, No. 57, Section 2, Zhongshan Rd, taiping District, Taichung City, Taiwan.
§ 九州情報大学, 福岡県太宰府市宰府 6 丁目 3-1, Kyushu Institute of Information Sciences, 6-3-1 Saifu, Dazaifu, Fukuoka, Japan.
¶ 北九州市立大学, 福岡県北九州市若松区ひびきの 1-1, The University of Kitakyushu, 1-1Hibikino, Wakamatsu, Kitakyushu, Fukuoka, Japan.

2 動的ランダム鍵生成器について

本章では Lin らの動的ランダム鍵生成器の構成と動作原理について概説する. これは乱数生成を行う 4 次元 Lorenz-Stenflo 方程式, 得られた乱数を用いて共通鍵を生成するハッシュ関数によって構成される. 動的ランダム鍵生成器の概要を図 1 に示す.

このシステムは方程式の同期を用いて, 送信側と受信側で共通鍵の生成を行うことが目的である. 両者はそれぞれの方程式を用いて写像を行う. 通常, 方程式のパラメータが共通であっても内部変数の初期値が異なっていると写像のたびに両者で異なる内部変数が得られる. 本システムでは内部変数の一致を目的として設計された同期信号を送信側から送り, 受信側で自身の方程式に加算する. これにより両者のパラメータが共通であれば受信側の内部変数は送信側のそれに漸近し最終的に両者は完全に同期する. 同期した内部変数をハッシュ関数に入力するため, 両者は同一の鍵を共有でき, 共通鍵暗号方式による秘密通信が可能となる.

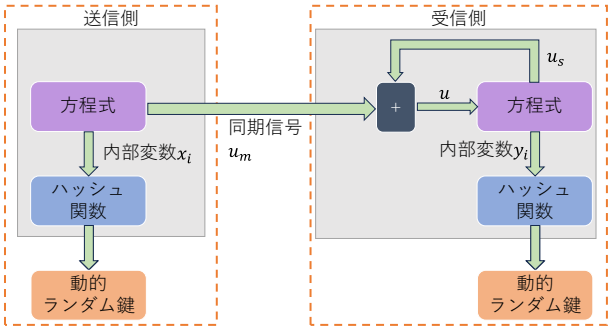

図 1: 動的ランダム鍵生成器の概要

3 4 次元 Lorenz-Stenflo 方程式について

本章では動的ランダム鍵生成器内に搭載されている方程式である 4 次元 Lorenz-Stenflo 方程式についてその式の内容, および送信側と受信側の差が漸近的に収束するための条件について紹介する.

動的ランダム鍵生成器内部にある 4 次元 Lorenz-Stenflo 方程式について一般化されたものを以下に示す. このと

き $\dot{x}_i(t)$ は微分値を示している．

$$\begin{aligned}\dot{x}_1(t) &= -ax_1(t) + ax_2(t) + \lambda x_3(t)\\ \dot{x}_2(t) &= dx_1(t) - x_2(t) - x_1(t)x_4(t)\\ \dot{x}_3(t) &= -cx_1(t) - x_3(t)\\ \dot{x}_4(t) &= x_1(t)x_2(t) - bx_4(t)\end{aligned} \tag{1}$$

このとき a, b, c, d, λ はパラメータであり，
$\dot{x}_1(t), \dot{x}_2(t), \dot{x}_3(t), \dot{x}_4(t)$ は内部変数の微分値を示している．微分値を用いる状態遷移式を式 (2) に示す．

$$x_i(t+1) = x_i(t) + \dot{x}_i(t) \times \delta t \tag{2}$$

送信側の 4 次元 Lorenz-Stenflo 方程式を式 (1), 受信側の 4 次元 Lorenz-Stenflo 方程式を式 (3) に示す．

$$\begin{aligned}\dot{y}_1(t) &= -ay_1(t) + ay_2(t) + \lambda y_3(t) + u_1(t)\\ \dot{y}_2(t) &= dy_1(t) - y_2(t) - y_1(t)y_4(t) + u_2(t)\\ \dot{y}_3(t) &= -cy_1(t) - y_3(t)\\ \dot{y}_4(t) &= y_1(t)y_2(t) - by_4(t)\end{aligned} \tag{3}$$

式 (3) の $u_1(t), u_2(t)$ は内部変数が同じ値をとるように設計された同期信号を示している．

送信側と受信側の差を $e_i = y_i - x_i$ とすると式 (4) のように示すことが出来る．

$$\begin{aligned}\dot{e}_1(t) &= -ae_1(t) + ay_2(t) + \lambda y_3(t)\\ &\quad - ax_2(t) - \lambda x_3(t) + u_1(t)\\ \dot{e}_2(t) &= de_1(t) - e_2(t) - y_1(t)y_4(t) + x_1(t)x_4(t) + u_2(t)\\ \dot{e}_3(t) &= -ce_1(t) - e_3(t)\\ \dot{e}_4(t) &= -be_4(t) + y_1(t)y_2(t) - x_1(t)x_2(t)\end{aligned} \tag{4}$$

ここで $u_1(t), u_2(t)$ は式 (5) のように構成されている．

$$\begin{aligned}u_1(t) &= -ay_2(t) - \lambda y_3(t) + ax_2(t) + \lambda x_3(t)\\ u_2(t) &= y_1(t)y_4(t) - x_1(t)x_4(t)\end{aligned} \tag{5}$$

これらの同期信号を受信側と送信側の差 $e_i(t) = y_i(t) - x_i(t)$ の式に代入すると式 (6) のようにできる．

$$\begin{aligned}\dot{e}_1(t) &= -ae_1(t)\\ \dot{e}_2(t) &= de_1(t) - e_2(t)\\ \dot{e}_3(t) &= -ce_1(t) - e_3(t)\\ \dot{e}_4(t) &= -be_4(t) + y_1(t)y_2(t) - x_1(t)x_2(t)\end{aligned} \tag{6}$$

$\dot{e}_1(t) = -ae_1(t)$ より $a > 0$ である前提のもと，
$\lim_{t\to\infty} e_1(t) = 0$ となる．$e_1(t) = 0$ となると，$\dot{e}_2(t) = -e_2(t)$ となることから $\lim_{t\to\infty} e_2(t) = 0$ となる．また，$\dot{e}_3(t) = -e_3(t)$ であるため $\lim_{t\to\infty} e_3(t) = 0$ となる．$e_1(t) = 0$ かつ $e_2(t) = 0$ となると，$y_1(t) = x_1(t)$, $y_2(t) = x_2(t)$ であるため $\dot{e}_4(t) = -be_4(t)$ より $b > 0$ である前提のもと，$\lim_{t\to\infty} e_4(t) = 0$ となる．

そのため全ての受信側の内部変数は同期信号を加算しながら写像を繰り返すと，送信側の内部変数との差がなくなるため写像を実行するごとに共通の鍵が生成される．

4 整数変換方法

3 章で示した 4 次元 Lorenz-Stenflo 方程式は連続的な値をとる変数で構成されている．これをコンピュータ上でシミュレートする際，従来の研究では倍精度浮動小数点数などが用いられてきた．しかし，1 章で述べた通り浮動小数点数演算は計算順序によって微小な誤差を生むため，数式的に等価な実装であっても異なる値となる場合がある．厳密な値の同期が要求される本システムにおいては通信の可否に関わるため問題となる．

この問題を解決するために本章では 4 次元 Lorenz-Stenflo 方程式の整数変換方式 [2] について紹介するとともに，整数変換を行った 4 次元 Lorenz-Stenflo 方程式を導出する．

4.1 整数変換方式の導出

式 (1) にて示されている 4DLS が取りうる内部変数の値の範囲は $-64 \leq x_i < 64$ に収まっている．整数変換を行う時に用いる演算精度 n としたとき，内部変数の範囲 $-64 \leq x_i < 64$ を整数値 $X_i \in [0, 2^n]$ にそのまま置き換える．変換式を式 (7) に示す．

$$x_i = \frac{128X_i}{2^n} - 64 \tag{7}$$

また，方程式内部の計算を整数値で行うことを目的としてパラメータの変換を行う．変換式を式 (8) に示す．

$$a = \frac{A}{2^n} \tag{8}$$

4.2 整数版 4 次元 Lorenz-Stenflo 方程式の導出

式 (7, 8) の変換を送信側の 4 次元 Lorenz-Stenflo 方程式である式 (1), 受信側の 4 次元 Lorenz-Stenflo 方程式である式 (3) に対して用い，整数変換を行う．

送信側の 4 次元 Lorenz-Stenflo 方程式である式 (1) に

整数変換を行った方程式を式 (9) に示す.

$$
\begin{aligned}
\dot{X}_1(t) &= \frac{-AX_1(t) + AX_2(t) + \Lambda X_3(t) - \Lambda 2^{n-1}}{2^n} \\
\dot{X}_2(t) &= \frac{DX_1(t) - 2^{n-1}D - X_2(t) + 2^{n-1} + 2^{n+6}X_1(t)}{2^n} \\
&\quad + \frac{2^{n+6}X_4(t) - 128X_1(t)X_4(t) - 2^{2n+5}}{2^n} \\
\dot{X}_3(t) &= \frac{-CX_1(t) - 2^{n-1}X_3(t) + 2^{n-1}C + 2^{2n-1}}{2^n} \\
\dot{X}_4(t) &= \frac{128X_1(t)X_2(t) - BX_4(t) - 2^{n+6}X_1(t)}{2^n} \\
&\quad + \frac{-2^{n+6}X_2(t) + 2^{n-1}B + 2^{2n+5}}{2^n}
\end{aligned}
\tag{9}
$$

受信側の 4 次元 Lorenz-Stenflo 方程式である式 (3) に整数変換を行った方程式を式 (10) に示す.

$$
\begin{aligned}
\dot{Y}_1(t) &= \frac{-AY_1(t) + AY_2(t) + \Lambda Y_3(t) - \Lambda 2^{n-1} + U_1(t)}{2^n} \\
\dot{Y}_2(t) &= \frac{DY_1(t) - 2^{n-1}D - Y_2(t) + 2^{n-1} + 2^{n+6}Y_1(t)}{2^n} \\
&\quad + \frac{2^{n+6}Y_4(t) - 128Y_1(t)Y_4(t) - 2^{2n+5} + U_2(t)}{2^n} \\
\dot{Y}_3(t) &= \frac{-CY_1(t) - 2^{n-1}Y_3(t) + 2^{n-1}C + 2^{2n-1}}{2^n} \\
\dot{Y}_4(t) &= \frac{128Y_1(t)Y_2(t) - BY_4(t) - 2^{n+6}Y_1(t)}{2^n} \\
&\quad + \frac{-2^{n+6}Y_2(t) + 2^{n-1}B + 2^{2n+5}}{2^n}
\end{aligned}
\tag{10}
$$

また, 同期信号である式 (5) に整数変換を行った方程式を式 (11) に示す.

$$
\begin{aligned}
U_1(t) &= AX_2(t) + \Lambda X_3(t) - \{AY_2(t) + \Lambda Y_3(t)\} \\
U_2(t) &= 2^{n+6}X_1(t) + 2^{n+6}X_4(t) - 128X_1(t)X_4(t) - 2^{2n+5} \\
&\quad - \{2^{n+6}Y_1(t) + 2^{n+6}Y_4(t) - 128Y_1(t)Y_4(t) - 2^{2n+5}\}
\end{aligned}
\tag{11}
$$

式 (9, 10, 11) を用いて状態遷移を行う式を式 (12) に, 写像を行う流れを図 2 に示す. このとき, $\delta t = \Delta T/2^{10}, \Delta T \in [0, 2^{10}]$, $\dot{X}_i(t) = \bar{X}_i(t)/2^n$ とする.

始めに初期値である $x_i(0), y_i(0)$ を整数値に変換する. 整数値に変換された初期値を用いて同期信号 U_m, U_s を求めたのちに整数変換を行った方程式を用いて次の内部変数を求める. これを繰り返し写像を行う.

$$
X_i(t+1) = X_i(t) + \dot{X}_i(t) \times \Delta T = \frac{2^{n+10}X_i(t) + \bar{X}_i(t)\Delta T}{2^{n+10}}
\tag{12}
$$

5 整数変換を行った場合の検証

本章では整数変換を行った 4 次元 Lorenz-Stenflo 方程式について検証を行った. 特に共通鍵生成の条件である

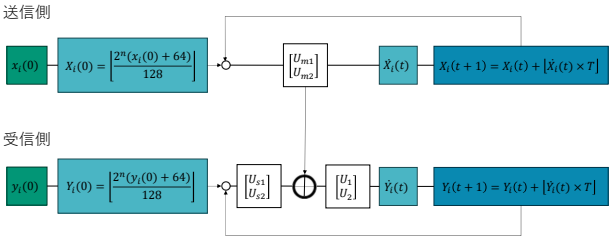

図 2: 整数変換を行ったローレンツ方程式の写像

内部変数の同期について結果を示す. 検証で用いたパラメータはいずれも $a = 11, b = 2.9, c = 5, d = 23, \lambda = 1.9$ とし, ステップ幅 $T = 0.01$ とした. また初期値は送信側を $x_1(0) = -1.0, x_2(0) = -3.0, x_3(0) = 2.0, x_4(0) = -5.0$, 受信側を $y_1(0) = -1.1, y_2(0) = -3.0, y_3(0) = 2.0, y_4(0) = -4.0$ とし, 整数変換を行った場合の演算精度および比較時の従来手法の仮数部のビット数 $n = 16$ としている.

5.1 式を整数変換した場合の値の推移

整数変換をした場合の内部変数の差 E_i が写像ごとに収束するか検証を行った. 結果を図 3 に示す. この結果は整数変換を行った場合でも差が収束するため鍵生成に活用可能であることを示している.

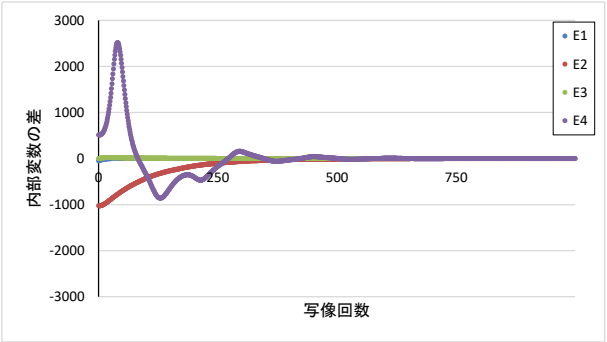

図 3: 整数変換をした場合の内部変数の差

5.2 計算方法の違いと同期

従来の手法では値の一致が困難となる組み合わせである計算の順序が異なる場合, 同期信号の計算の流れが異なる場合について整数変換が有効であるか検証を行った. 従来手法と提案手法を対象とした写像回数ごとの値の一致について結果を記す.

5.2.1 計算の順序について

送信側と受信側で内部変数の計算式の項の順序が異なる場合の計算結果について示す. 本研究では式 (3) の $y_2(t)$ の式の第 2 項と第 3 項を入れ替えた場合である式式 (13) を従来の手法を用いた場合と整数変換を行った場合の受信側の方程式としたときの同期状況について調

査を行った.

$$
\begin{aligned}
\dot{y}_1(t) &= -ay_1(t) + ay_2(t) + \lambda y_3(t) + u_1(t) \\
\dot{y}_2(t) &= dy_1(t) - y_1(t)y_4(t) - y_2(t) + u_2(t) \\
\dot{y}_3(t) &= -cy_1(t) - y_3(t) \\
\dot{y}_4(t) &= y_1(t)y_2(t) - by_4(t)
\end{aligned}
\tag{13}
$$

従来手法の結果を図 4 に, 提案手法の結果を図 5 に示す. この図は各内部変数 E_i と同期信号 U_i について写像回数ごとの同期状況について, 送信側と受信側が同期している状態に対応する箇所を表示している. ここから提案手法では計算順序は値の同期に依存していないことがわかる.

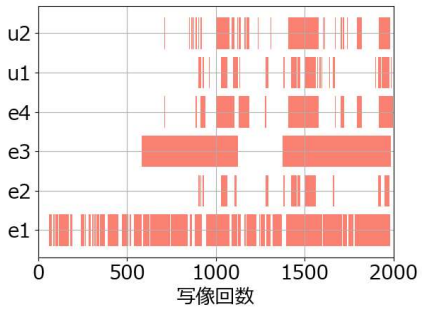

図 4: 計算順序を入れ替えた場合の従来手法

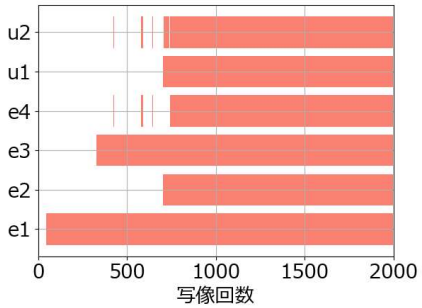

図 5: 計算順序を入れ替えた場合の整数変換後

5.2.2 同期信号の計算の流れについて

コンピュータ上での同期信号の計算方法と同期状況の関係について示す.

提案手法を用いた時同期信号の計算方法によって値の同期状態が異なる. 本研究では送信側と受信側でそれぞれ計算される同期信号の値を一度まとめ, 受信側に加算する場合である式 (14) と微分値を計算する時に全ての値を換算する場合である式 (15) について検証を行った.

$$
\begin{aligned}
\dot{y}_1(t) &= -ay_1(t) + ay_2(t) + \lambda y_3(t) \\
&\quad + ax_2(t) + \lambda x_3(t) - ay_2(t) - \lambda y_3(t) \\
\dot{y}_2(t) &= dy_1(t) - y_2(t) - y_1(t)y_4(t) \\
&\quad - x_1(t)x_4(t) + y_1(t)y_4(t) \\
\dot{y}_3(t) &= -cy_1(t) - y_3(t) \\
\dot{y}_4(t) &= y_1(t)y_2(t) - by_4(t)
\end{aligned}
\tag{14}
$$

$$
\begin{aligned}
u_1(t) &= -ay_2(t) - \lambda y_3(t) + ax_2(t) + \lambda x_3(t) \\
u_2(t) &= y_1(t)y_4(t) - x_1(t)x_4(t) \\
\dot{y}_1(t) &= -ay_1(t) + ay_2(t) + \lambda y_3(t) + u_1(t) \\
\dot{y}_2(t) &= dy_1(t) - y_2(t) - y_1(t)y_4(t) + u_2(t) \\
\dot{y}_3(t) &= -cy_1(t) - y_3(t) \\
\dot{y}_4(t) &= y_1(t)y_2(t) - by_4(t)
\end{aligned}
\tag{15}
$$

結果をそれぞれ図 6, 図 7 に示す. 数式的にはどちらも等価であるが式 (15) を用いる場合のみ値が一致している.

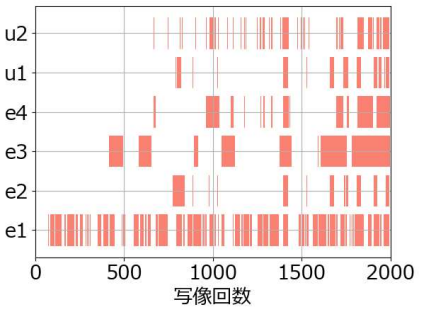

図 6: 同期信号を同時に計算する場合の従来手法

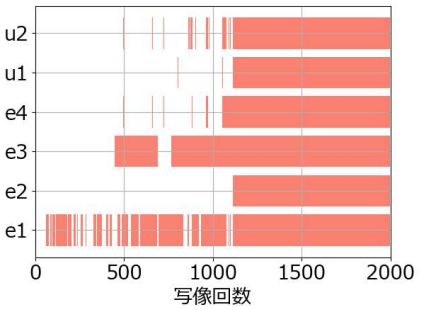

図 7: 同期信号を先に計算する場合の従来手法

提案手法での結果をそれぞれ図 8, 図 9 に示す. ここから提案手法では計算順序を入れ替えた場合と同様に同期信号の計算方法に依存せずに値の同期することがわかる.

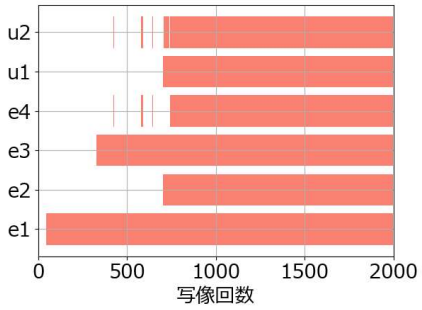

図 8: 同期信号を同時に計算する場合の提案手法

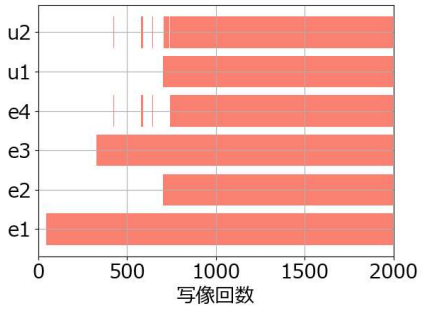

図 9: 同期信号を先に計算する場合の提案手法

6　おわりに

本研究では Lin らの動的ランダム鍵生成器での利用を想定し, 4 次元 Lorenz-Stenflo 方程式の整数変換手法を提案した. 従来手法が抱える浮動小数点演算に起因した, 計算順序などの実装方法に計算結果が依存している問題に対し, 提案手法では計算結果が一意に定まることから安定した同期が可能であることを実証した.

これにより動的ランダム鍵生成器の応用範囲を広げることが期待される.

参考文献

[1] C. Lin, G. Hu, J. Chen, J. Yan and K. Tang, “Novel design of cryptosystems for video/audio streaming via dynamic synchronized chaos-based random keys,” Multimedia Systems, Vol. 28, pp.1793-1808, 2022.

[2] 荒木, 髙市, 顔, 宮﨑, 上原, “擬似乱数生成に向けたローレンツ方程式の整数化に関する研究, ” 信学技報, Vol. 124, No. 147, IT2024-22, pp. 48-53, 2024.