デジタルスパイクマップの進化的動作最適化とFPGA実装

デジタルスパイクマップの進化的動作最適化と FPGA 実装

Evolutionary dynamics optimization and FPGA based implementation of digital spike map

原田朋樹
Tomoki HARADA
指導教員　齋藤利通

法政大学大学院理工学研究科電気電子工学専攻修士課程

In this thesis, we consider optimization and FPGA based implementation of digital spike maps. First, the dynamics of spike-trains is visualized by a digital spike map. The map is defined on a set of points and is represented by a characteristic vector of integers. Second, we introduce a simple evolutionary algorithm for optimization of digital spike maps. We use autocorrelation function as a cost function. Third, in order to implement the digital spike map, we introduce a digital spiking neuron. Repeating integrate-and-fire behavior between a periodic base signal and constant threshold, the neuron can out put various periodic spike-trains. The digital spike maps are implemented in an FPGA board and typical spike-trains are confirmed experimentally.

***Key Words** : digital spike maps, simple evolutionary algorithm, FPGA*

１．はじめに

本論文ではデジタルスパイクマップの最適化とハードウェア実装について考察する．Dmap は有限の点の集合上で定義されているデジタル力学系である[1][2]．Dmap はロジスティックマップ[3] のようなアナログ 1 次元マップのデジタル版とみなすことができる．Dmap はカオスな現象が生じることなく，様々な周期スパイク列を生成する．応用例には，デジタル通信[4][5]，セントラルパターンジェネレータ(CPG)[6]，時系列近似[7]などがある．また，Dmap は FPGA でのハードウェア実装も可能である[7]．

所望のスパイク列生成する Dmap を最適化するために，簡素な進化的アルゴリズム(SEA)[8]を用いる．SEA は貪欲法に基づいた進化的アルゴリズム(EA)[9]の一種で，ヒューリスティックな最適化アルゴリズムとして知られている．SEA は個体の複製と突然変異を行い，評価値が高い個体を記憶する．記憶した個体を次の世代に引き継いでいくことで，最適値を探索するアルゴリズムである．本論文では，評価関数として，自己相関関数を用いることにする．

次に SEA によって最適化された Dmap をハードウェア実装するため，デジタルスパイキングニューロン(DSN)[8]を定義する．DSN はスパイキングニューロンモデルを参考に考案されたものである．周期的なベース信号と一定のしきい値の間で積分と発火を繰り返すことで，様々な周期スパイク列を生成する．DSN を用いることで所望の Dmap をハードウェア実装できる．典型例として SEA で最適化した超安定状態の Dmap を FPGA でハードウェア実装をし，周期スパイク列を実験的に観測する．

２．デジタルスパイクマップ

図 1 にデジタルスパイクマップ(Dmap) とスパイク列を示す．τ は離散時間，τ_n はn番目のスパイク位置，N はクロック周期である．このスパイク列は，1 クロック周期に 1 本のスパイクが立ち，$\theta_n = \tau_n \bmod N$が成り立つ．Dmap は以下の差分方程式で定義される．

$$\theta_{n+1} = F(\theta_n),\ \theta_n \in \{1,2,\cdots,N\} \equiv L_N \qquad (1)$$

Dmap は以下に示す特性ベクトルd によって特徴づけられる．

$$d \equiv (d_1,\cdots,d_N), F(i) = d_i \in \{1,\cdots,N\} \qquad (2)$$

例として図 1 の Dmap は特性ベクトル
$d = (6,5,7,2,7,2,4,1)$によって特徴づけられている．ス

パイクの初期位置が決定されると，スパイク位相系列 $S=(\theta_1,\cdots,\theta_p)$が出力される．これによって以下の式であらわされるスパイク列が生成される．

$$Y(\tau)=\begin{cases}1 & \text{for } \tau=\tau_n \\ 0 & \tau\neq\tau_n\end{cases} \quad \tau_n=\theta_n+(n-1) \qquad (3)$$

n 番目のスパイクはn 番目の 1 周期間に生成される $\tau_n\in[n-1,n)$．また，$p\in L_N$ を満たす点は周期点(PEP)と呼ばれており，PEP は特性ベクトルでは赤字，図 1 では赤い点で示している．周期点によって周期軌道(PEO)が生成される．また，図 1 における周期点以外の点(黒)は数回の写像で周期軌道に落ち込む．この点を E 周期点と定義する．周期軌道以外のすべての点が EPP の場合，その Dmap は安定状態である.また，1 度の写像で周期軌道に落ち込む点を直接 E 周期点と呼び，すべての周期点以外の点が直接 E 周期点の Dmap は超安定状態である．超安定状態の Dmap はハードウェア実装が可能である．

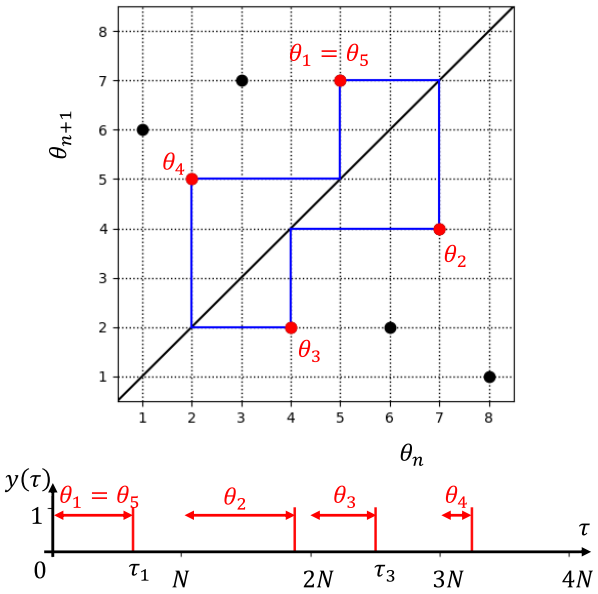

図 1　デジタルスパイクマップと周期スパイク列

3．簡素な進化的アルゴリズム

Dmap の最適値探索方法として簡素な進化的アルゴリズム(SEA) を提案する．Dmap はN^N通り存在するため全探索すると膨大な計算コストがかかる．そこで SEA を用いることでN の多項式で計算することが可能になる．本論文では大規模な Dmap を扱うため，さらに計算コストを減らすために乱数で個体の生成をする．また，本論文では評価関数に自己相関の第二ピークを用いた．周期軌道p を持つ基本クロック周期Nの自己相関関数を以下のように定義する．

$$F_c(d)=max_s R_{yy}\geq 1, R_{yy}(s)=\sum_{\tau=0}^{pN} S(\tau)S(\tau+s) \qquad (4)$$
$$\text{for} \quad s\in\{1,\cdots,pN-1\}$$

SEA において世代g における個体を次のように定義する．

$$\delta^i(g)=\left(\delta_1^i(g),\cdots,\delta_N^i(g)\right), i\in\{1,2,\cdots,Kg(g)\} \qquad (5)$$

$K(g)$ は世代g における個体数，個体数上限を$K_{g_{max}}$，世代数上限をg_{max}，変化上限をc_{max}，候補数をβ_{max}とする．全ての個体を評価関数$F_c\left(\delta^i(g)\right)$で評価し，世代$g$ における最良値をグローバルベスト$G_b(g)$ とする．SEA は 4 つのステップで定義する．

Step1(初期化)$g=0$，$K_g(0)=1$ とする．基本周期N を設定する．階段状の周期軌道$p=N/2$ を持つ初期個体$\delta^1(0)$を与える.$G_b(0)=F_c\left(\delta^1(0)\right)$とする．

Step2(突然変異)

Step2-1(周期点)i 番目の個体$\delta^i(g), i\in 1,2,\cdots,K_g(g)$の周期点を 1 つの異なる点をランダムに選択して写像するように変化を与える．候補の数として$\beta_{max}\times K(g)$個得られる．候補を$\beta^j(g), j=1\sim\beta_{max}\times K(g)$ として定義する．

Step2-2(周期点以外の点) 全ての候補に対して周期点以外の点を 1 つランダムで選択し，ランダムで選択した位置に写像するように変化を与える．これを変化上限c_{max}回行う．

Step3(候補を評価) Step2 で行った所望の周期である全ての個体を評価関数$F_c(d)$ によって評価して，$G_b(g)$を更新する．

$$G_b(g)=\begin{cases}F_c\left(\beta^j(g)\right) & \text{if} \quad F_c\left(\beta^j(g)\right)<G_b(g) \\ G_b(g) & \text{otherwise}\end{cases} \qquad (6)$$

世代g において評価値が高い候補を次世代の個体として記憶する．この個体数が$K_{g_{max}}$個を超えた場合，個体群の中からランダムに$K_{g_{max}}$ 個を次世代の個体として選択する．

Step4(終了判定)$G_b(g)$が最良値であれば終了．最良値でなかった場合，世代を更新して，Step2 に戻る．これを世代上限g_{max}まで繰り返す.$g\in\{0,1,2,\cdots,g_{max}\}$

4．数値実験

SEA に用いる初期の Dmap を特性ベクトルで以下に示す．

d = (2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17,
18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32,
33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47,
48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62,
63, 64, 1, 32, 43, 13, 39, 39, 64, 111, 44, 33, 16, 25, 122,
57, 26, 1, 49, 54, 97, 98, 63, 9, 71, 55, 25, 54, 98, 34,86,
111, 24, 7, 92, 86, 72, 98, 87, 41, 101, 77, 72, 73, 46,108,
5, 100, 77, 100, 94, 13, 41, 34, 117, 33, 124, 23, 113, 1,
48, 31, 33, 28, 93, 111, 58)

図 3 に$g = 100$ の Dmap をそれぞれ示す．$g = g_{max}$ で，Dmap の特性ベクトルは以下のようになった．

d = (42, 115, 79, 40, 56, 124, 17, 5, 62, 36, 30, 80, 82, 66, 78, 33,122, 23, 95, 105, 12, 59, 87, 35, 118, 60, 128, 14, 31, 78, 37, 19, 78, 98, 5, 103, 23, 108, 75, 75, 4, 97, 127, 39, 64, 29, 63, 81, 7, 88, 21, 104, 103, 121, 121, 43, 40, 9, 51, 51, 9, 110, 96, 27, 29, 57, 47, 124, 63, 125, 68, 15, 2, 90, 2, 28, 28, 42, 102, 74, 47, 45, 125, 97, 107, 94, 108, 74, 128, 126, 3,107, 72, 82, 105, 120, 19, 22, 64, 117, 125, 31, 44, 109, 86, 52,93, 98, 14, 35, 90, 123, 91, 33, 62, 68, 46, 61, 96, 123, 53, 120, 85, 9, 113, 106, 81, 101)

このとき,$F_c(d) = 4$ であった.設定した各パラメータは以下である．

- 次元数N = 128
- 周期p = 64
- 世代数上限$g_{max} = 100$
- 候補数$\beta_{max} = 100$
- 個体数上限$K_{g_{max}} = 5$
- 変化数上限$c_{max} = 64$

また，図 4 にグローバルベスト$G_b(g)$の推移を表すグラフを示す．図 4 の黒色の推移は Step2-2 を省いたものである．これを SEA①とした．青色の推移は Step2-2 の遺伝操作をおこなったものである．これを SEA②とした．この図 4 から Step2-2 はトラップを回避しやすくする遺伝操作である．

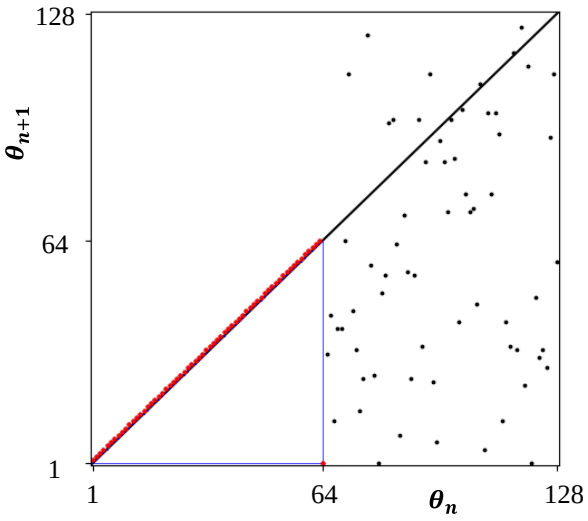

図 2　初期個体の Dmap($N = 128, p = 64$)

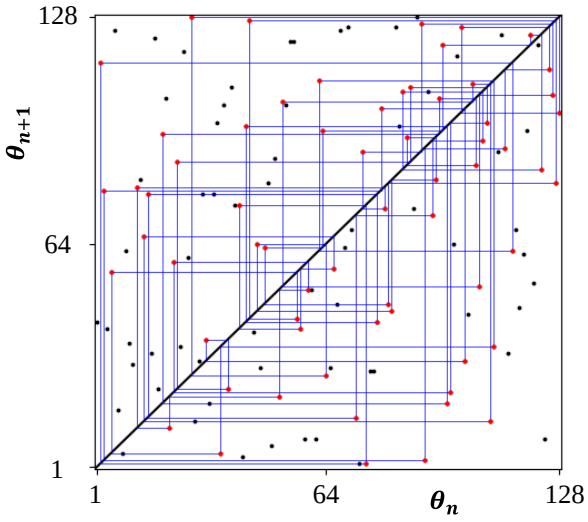

図 3　数値実験後($g = g_{max}$)の Dmap($N = 128, p = 64$)

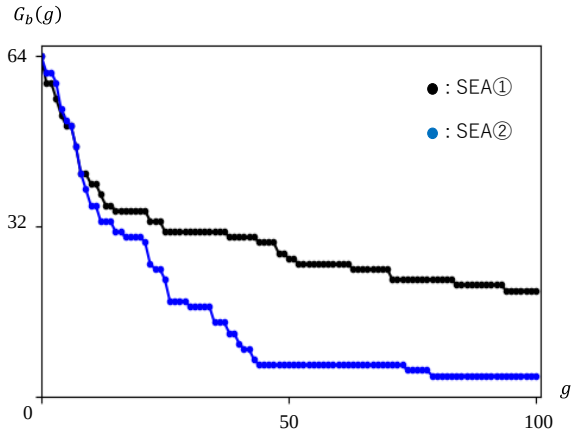

図 4　各世代gにおけるグローバルベスト$G_b(g)$の推移

５．FPGA 実装

超安定状態の Dmap は FPGA でハードウェア実装することができる．ハードウェア実装する手法としてデジタルスパイキングニューロン(DSN) の動作原理を用いる．τ は離散時間で，$x(\tau)$は離散状態変数である．x は時間とともに増加していき，しきい値N_x に達すると発火してベース信号$b(\tau)$にリセットされる．この動作を繰り返すことにより，DSN はスパイク列$y(\tau)$ を出力する．

積分

$$\begin{cases} x(\tau+1) = x(\tau) + 1 \\ y(\tau) = 0 \end{cases} \quad \text{if} \quad x(\tau) < N_x \tag{7}$$

自己発火

$$\begin{cases} x(\tau+1) = b(\tau) \\ y(\tau) = 1 \end{cases} \quad \text{if} \quad x(\tau) = N_x \tag{8}$$

ただし，$x(\tau) \in \{0,1,\cdots,N_x\}$ で，$b(\tau)$ は周期$N_p = N$ のベース信号である．簡単にするため，ベース信号に次の条件を与える．

$$\tau - 2N_p + 1 \le b(\tau) - N_x \le \tau - N_p \qquad (9)$$

このときN_xを以下のように定義する.

$$N_x = 2N_p - 1, 0 \le a_i - i \le N_p, i \in \{1,2,\cdots,N_p\} \qquad (10)$$

ただし，a_iはDSNの配線ベクトルと呼ばれており，ベース信号$b(\tau)$を特徴づける．Dmapの特性ベクトルが$d = \left(d_1,\cdots,d_{N_p}\right)$と与えられると，対応するDSNの配線ベクトルとの関係は以下のように決定される.

$$a = \left(a_1,\cdots,a_{N_p}\right), a_i = N_p - (d_i - i), i \in \{1,\cdots,N_p\} \quad (11)$$

図1の場合，配線ベクトルは以下のようになる.

$$d = (6,5,7,2,7,2,4,1)$$
$$\downarrow$$
$$a = (3,5,4,10,6,12,11,15)$$

ベース信号に依存して，DSNは様々なスパイク列を生成する.

$$\theta_{n+1} = F(\theta_n) = f(\theta_n) \bmod N_p$$
$$f(\tau_n) = \tau_n - b(\tau_n) + N_x + 1 \qquad (12)$$

Verilog HDLを用いてDSNの回路を記述した．この回路を用いることで様々なDmapをハードウェアで実現することができる．その結果が図5に示すFPGAの波形である．これは図3を超安定状態にしてハードウェア実装したもので,$N = 128, p = 64$となっている．また，このスパイク列は自己相関の第二ピークが低い.

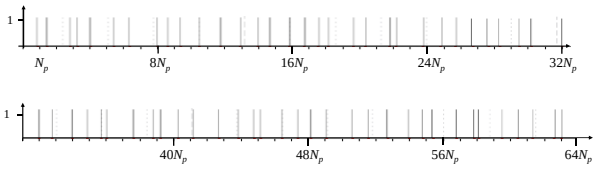

図5　FPGAで観測した周期スパイク列
$(N = 128, p = 64)$

６．まとめ

Dmapの単目的最適化とFPGA実装について考察をした．Dmapは有限個の点の集合上で定義されているデジタル力学系であり，定常状態では必ず周期スパイク列を生成する．特性ベクトル依存して，様々な周期スパイク列が生成される．Dmapは定義域が大きくなると,特性ベクトルのパターン数が指数関数的な増加をするため全探索による最適化が困難である．そこで本論文では，SEAを用いて探索コストの削減を試みた．また，大規模なDmapを最適化するために，SEAのアルゴリズム内の個体生成に乱数を用いた．評価関数はDmapを周期スパイク列の自己相関の第二ピークとした．数値実験を通してこの評価関数に対する最適な近似解を得ることができた.また，世代が更新されるにつれて,グローバルベストが良い値を取るようになっていく様子を図に示した.この図でStep2-2がトラップを回避しやすくする遺伝操作であることがわかる.

次にDmapのハードウェア実装にはDSNの動作を参考にした．DSNはベース信号と一定のしきい値の間で積分と発火の動作を繰り返すことで，様々な周期スパイク列が生成される．DSNではDmapが生成する周期スパイク列のハードウェア実装が可能となる．Verilog HDLを用いてDSNの回路の記述を行いた．また，FPGAを用いて大規模な基本周期をもつスパイク列の出力をした.今後の課題として他の評価関数での最適化アルゴリズム開発やDmapの結合系の最適化などが挙げられる.

参考文献

1) H. Torikai, H. Hamanaka, and T. Saito, Reconfigurable Spiking Neuron, and Its Pulse-Coupled Networks: Basic Characteristics and Potential Applications, IEEE Trans. Circuits Syst. II, 53, 8, pp. 734-738, 2006.
2) H. Torikai, A. Funew, and T. Saito, Digital Spiking Neuron and Its Learning for Approximation of Various Spike-Trains, Neural Networks, 21, pp. 140-149, 2008.
3) E. Ott, Chaos in Dynamic Systems, Cambridge, 1993.
4) T. Iguchi, A. Hirata, and H. Torikai, Theoretical and Heuristic Synthesis of Digital Spiking Neurons for Spike-Pattern-Division Multiplexing, IEICE Trans. Fundamentals, E93-A, 8, pp. 1486-1496, 2010. 49
5) N. F. Rulkov, M. M. Sushchik, L. S. Tsimring, and A. R. Volkovskii, Digital Communication Using Chaotic-Pulse-Position Modulation, IEEE Trans. CAS-I, 48, 12, pp. 1436-1444, 2001.
6) A. Lozano, M. Rodriguez, and R. Roberto Barrio, Control Strategies of 3-cell Central Pattern Generator via Global Stimuli, Sci. Rep. 6, 23622; doi: 10.1038/srep23622, 2016.
7) H. Uchida, Y. Oishi and T. Saito, A Simple Digital Spiking Neural Network: Synchronization and Spike-Train Approximation, Discrete Contin. Dyn. Syst. Ser. S., doi: 10.3934/dcdss.2020374, 2020.
8) W. Ertel, Introduction to Artificial Intelligence, Springer, 2009.
9) A. P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, Willey, 2005.