国際研究論叢　29（2）：47～62，2016

セル・オートマトンによる閉鎖的社会構造モデルの分析

安達　康生[*1]　　安高真一郎[*2]　　植松　康祐[*3]

Analysis of Closed Social Structure Models Using the Cellular Automaton

Yasuo Adachi[*1]　　Shinichiro Ataka[*2]　　Koyu Uematsu[*3]

Abstract

The cellular automaton was created by John Von Neumann and Stanislaw Marcin Ulam. What Von Neumann was interested in was a machine which could replicate itself and the first such machine in the logical world was the Neumann cellular automaton. Since then, the cellular automaton has been expanded to many fields such as biology, history, and complex systems. This paper suggests closed social structure models with the method based on the cellular automaton. We succeeded in finding some interesting facts using simulations. The results gained can be applied to models such as closed social structures nowadays or the existence of living creatures.

キーワード

セル・オートマトン、シミュレーション

Key Words

Cellular Automaton, Simulation

第１章　セル・オートマトンとライフゲーム

セル･オートマトン（cellular automaton，CA）の概念は，1940年代にJohn Von NeumannとStanisław Marcin Ulamによって作られた．John Von Neumannは，20世紀を代表する大天才の一人であることは，誰もが認めるところである．世界最初のプログラム内蔵方式のコンピュータEDVAC（Electronic Discrete Variable Automatic Computer）の開発に関わった．その功績から，現在使われているコンピュータはノイマン型と呼ばれている．彼の業績は，数学分野では集合論や関数解析，物理学分野では量子力学における新しい数学

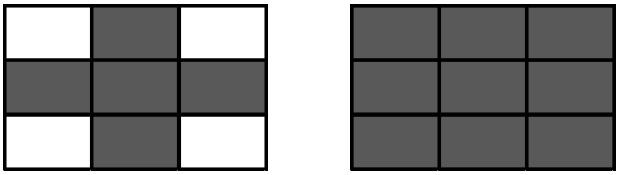
＊１　あだち　やすお：大阪国際大学グローバルビジネス学部准教授〈2015.9.25受理〉
＊２　あたか　しんいちろう：大阪国際大学グローバルビジネス学部准教授
＊３　うえまつ　こうゆう：大阪国際大学グローバルビジネス学部教授

表現，経済学においてはゲーム理論，そして，複雑系分野でのセル・オートマトンなどの現在での基礎科学となっている新しい領域を作り出した.

Stanisław Marcin Ulam は，ポーランド出身の数学者で，数学分野で多くの貢献をしているが，水素爆弾の開発で中心的な役割を果たしたことで知られている. 1943年に，Ulam は Neumann の招きで，マンハッタン計画に参加し，その後もロスアラモス国立研究所で研究を続けた．その時期に，二人はセル･オートマトンの概念を作り上げることになった.

Neumann が興味を持っていたのは，機械が自分自身と同一の機械を作れることであった．更に，その機械が自分自身より複雑な機械を製造して，機械が際限なく進化することが可能であるかということを考えていた. Ulam は Neumann に自己再製機械を解析するための抽象的な宇宙を作ってはどうかと勧めたのが始まりであった(1).

セル・オートマトンを簡単に説明すれば，Excel のように広いセル・シート上でのセルに，ある一定のルールを与えて時間の経過を観察するものである．各セルは，次の時刻にはすべてのセルが同期して新しい状態に変化する. それぞれの時刻をセル･オートマトンでは，世代（generation）と呼んでいる．次の世代のセル状態は，現時点での自分自身の状態と自分を取り巻く近傍（neighborhood）にあるセルの状態によって決定される．Neumann は，セルの状態を29種類に決め，28種類の状態は単純な機械部品に相当し，残りの１つを未開の空状態とした．そして，そのセルに辺で接する上下左右を近傍（図１．ノイマン近傍）とした．その後，Edward F. Moore は，点で接する斜めを含めた近傍（図１．ムーア近傍）でのシミュレーションを提案した.



ノイマン近傍　　ムーア近傍

図１．ノイマン近傍とムーア近傍

Neumann のセル宇宙は，無限に広いチェス盤と見なすことができる．プレーヤーは，そのチェス盤に28種類の駒を配置してゲームをスタートさせる．駒を置いていないところは空として，その後一定の規則に従って，自動的に刻々と変化する様子を眺めれば良い. Neumann は，このセル宇宙の中で自己再製できるような最初のパターンの存在を証明したことにより，論理的な世界で自己再製機械が作れることを示した(2)．上記で説明したセル・オートマトンは，正方形のセル集合であったが，その形を５角形や６角形としたものや，３次元に拡張したモデルなども研究されている.

セル･オートマトンは，様々な研究分野に影響を与えた. 生物学においては，生命を持った組織活動を化学と物理学に帰着できるかどうかが長年の課題であった．仮想的なセル・オートマトンによって，非常に複雑な細胞分裂などの生命活動を科学的に説明できる可能性を与えたと言われている. また，古代ローマ帝国が膨大な領土を拡大したにも関わらず，滅亡したような歴史的な国家の成立過程や滅亡などが，セル・オートマトンによって説明

される．R.Axelrod の「文化伝播モデル」では，文化には拡散する特性を持つが，１つには収斂しないことを示した．このモデルは，現代のボーダレスな情報化時代にあっても，ある特定の文化が消滅することや全世界を覆い尽くすことはないことを示唆する興味深い結果である[(3)(4)]．最近では，渋滞学にセル・オートマトンを応用した研究が，交通渋滞に効果を発揮している．車や人が渋滞する現象は，セルに時間の経過に従った移動のルールを与えることによってシミュレーションすることが可能である．これまでの渋滞学の研究では，非常に複雑なシミュレーションを行ってきたにも関わらず，有効な対策は見いだせてなかった．単純なセル・オートマトンが現象の本質を明確にしたことにより，待ち行列理論などを適応した有効な対策が打ち出されて，その結果が注目されている[(5)]．

ここでセル・オートマトンを利用したシミュレーションモデルであるライフゲームを紹介する．ライフゲームは，1970年にイギリスのケンブリッジ大学の数学者 John Horton Conway が作成したコンピュータゲームである．1974年に「Time」誌が「ライフゲームの大群が数百万ドルの貴重なコンピュータ時間を食ってしまっている」と苦言を呈したこともあった[(1)]．ライフゲームは，家庭用のコンピュータで楽しめる単純なゲームのように見えるが，Neumann が考えていた自己再製可能なものが構成できた．Conway は，２次元で２状態（０と１）のセル・オートマトンで作られたライフゲームの宇宙の中に，我々が生きる世界の変化や複雑性が凝縮されていることを示した．ここで，ライフゲームのルールを説明する．現状のセルが生存（１）しているとき，周りの８個のセル（ムーア近傍）の中で生存している個数が２個または３個のときには，次の世代では生存し，それ以外ならば死滅する．現状のセルが死滅（０）しているとき，周りの８個のセル（ムーア近傍）の中で生存している個数が３個のときには，次の世代では生存し，それ以外ならば死滅する．これだけの単純なルールである．たとえば，図２の初期状態（step ０）から始まったセルは，２世代目で死滅することになる．

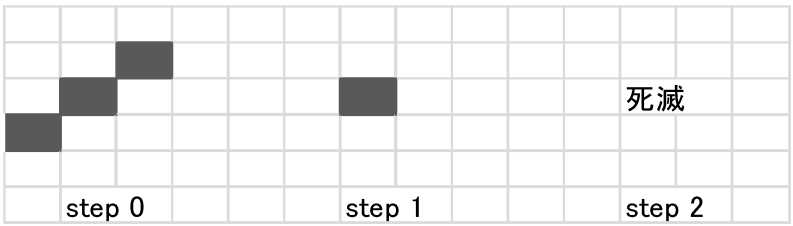


図２．セルの死滅例

また，図３のような初期状態からスタートすれば，２世代目（step ２）で再生されて循環する．

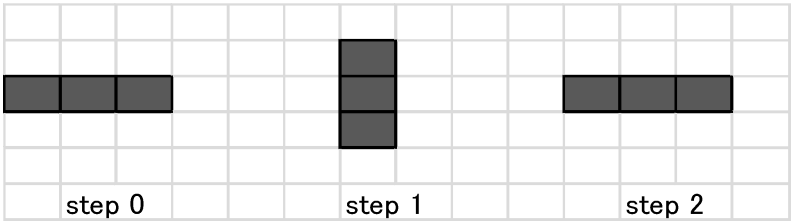


図３．セルの循環例

様々なシミュレーションによって，再生されるパターンや無限に成長するパターンが多数発見されている．また，初期状態から全く変化しない配置（エデンの園）が存在することが数学的に証明されるなど多数の研究成果がある．

本論文では，セル･オートマトンのルールを円周上に限定したモデルを提案する．主体を円周上限定した理由は，閉鎖的な社会構造をモデル化するときに，主体が両隣だけに影響して変化する単純構造が適切であると考えたからである．生物の生存や死滅を単純化すれば，遠く離れた生物の影響は受けず，自分と隣接する生物だけの影響を受ける．過密状態や孤立状態にいると死滅し，共存関係が保てると生き残ることができるという単純なルールとした．我々の興味は，どのような初期パターンが生存し，最終的な生存状態を知ることである．このシミュレーションを通して，様々な数学的な性質を発見することができたことは大きな成果である．

第２章　円周上でのセル・オートマトンモデル

３個以上の主体（セル）を円周上に配置するモデルを提案する．生存しているときは１，死滅しているときは０とする．ルールは，自分自身を含めて両隣で，生存しているセルの数が２個のときは，次世代では生き残る．それ以外，すなわち，生存しているセルの数が０個，１個，３個のときは，次世代では死滅する．これらを数学的に記述すれば，次の様になる．初期列［ $X_1(0), X_2(0), \cdots, X_n(0)$ ］ $X_k(0) \in \{0, 1\}$，（ k=1, 2, ･･･, n ），　n≧3
を円周上に配置する．これらの初期列は，次の規則に従ってステップ（generation）を進める．

$$X_k(i+1)=\begin{cases} 1, \text{ if } X_{k-1}(i)+X_k(i)+X_{k+1}(i)=2 \\ 0, \text{ otherwise} \end{cases} \quad (k=2, 3, \cdots, n-1)\ (i=0, 1, 2, \cdots)$$

ただし，k=1 のときは

$$X_1(i+1)=\begin{cases} 1, \text{ if } X_n(i)+X_1(i)+X_2(i)=2 \\ 0, \text{ otherwise} \end{cases} \quad (i=0, 1, 2, \cdots)$$

k=n のときは

$$X_n(i+1)=\begin{cases} 1, \text{ if } X_{n-1}(i)+X_n(i)+X_1(i)=2 \\ 0, \text{ otherwise} \end{cases} \quad (i=0, 1, 2, \cdots)$$

［ 定義 ］

初期列　［ $X_1(0), X_2(0), \cdots, X_n(0)$ ］に対して，すべての整数 i が
［ $X_1(i), X_2(i), \cdots, X_n(i)$ ］≠［ 0, 0, ･･･ , 0 ］ならば，この初期列を残存列と呼び，
［ $X_1(i), X_2(i), \cdots, X_n(i)$ ］＝［ 0, 0, ･･･ , 0 ］となる整数 i が存在するときを死滅列と呼ぶことにする．■

当然，すべての初期列は，残存列または死滅列に分類される．

［ Example ］

１を●（生存）０を●（死滅）と表し，n=5 のとき，初期列［ 1, 1, 1, 0, 0 ］を円周上に配置する．各点（セル）は上記の規則に従って，図４のような動きをして，３ステップ目で死滅することになる．すなわち，初期列［ 1, 1, 1, 0, 0 ］は死滅列であることがわかる．

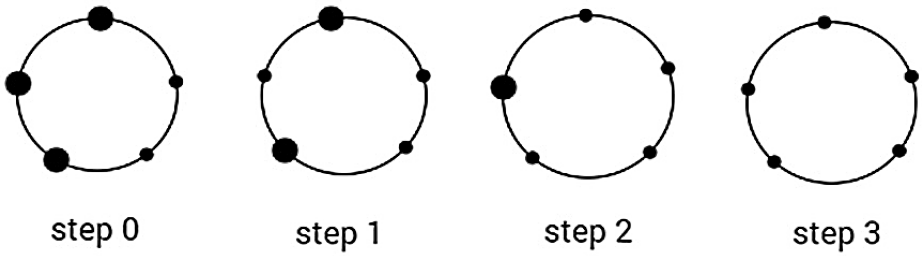


図４．n=5 のときの死滅列

n=5 のときの実験を行ってみると，ほとんどの初期列は死滅列となることがわかる．実は，残存列となる場合は，初期列［ 1, 1, 0, 0, 0 ］だけであることを確認できる．この初期列は，不動点となりステップを重ねても，動くことがない．

我々は，どのような初期列が残存列となるかについて調べた．

当然，n の数，すなわち円周に配置するセルの数が多くなれば非常に複雑化することは予想される．上記の規則は，n≧3に対して定義されているので，順次確認を行った．

（ｉ）n=3のとき，残存列は存在しない．

（ii）n=4のとき，図５に示す３つの初期列が残存列となる．

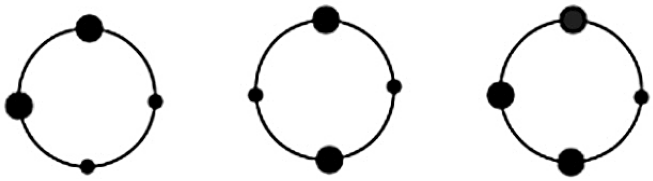


図５．n=4 のときの残存列

左側の初期列［ 1, 1, 0, 0 ］は，n=5のときに確認したように不動点である．真中と右側の初期列［ 1, 0, 1, 0 ］と［ 1, 1, 1, 0 ］は循環するが，円順列であると考えると形は変わらない．

（iii）n=5のときは，example で確認したように残存列は図６の１つだけである．

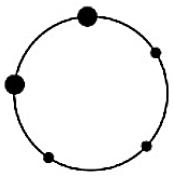


図６．n=5 のときの残存列

（iv）n=6のときは，残存列は図７に示す２つだけである．

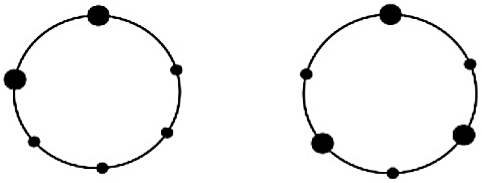


図７．n=6 のときの残存列

（ｖ）n=7のときは，残存列は図８に示す２つだけである．

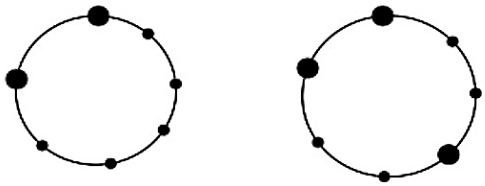


図８．n=7 のときの残存列

（vi）n=8のときは，急に複雑となり，残存列は図９に示す７つとなる．

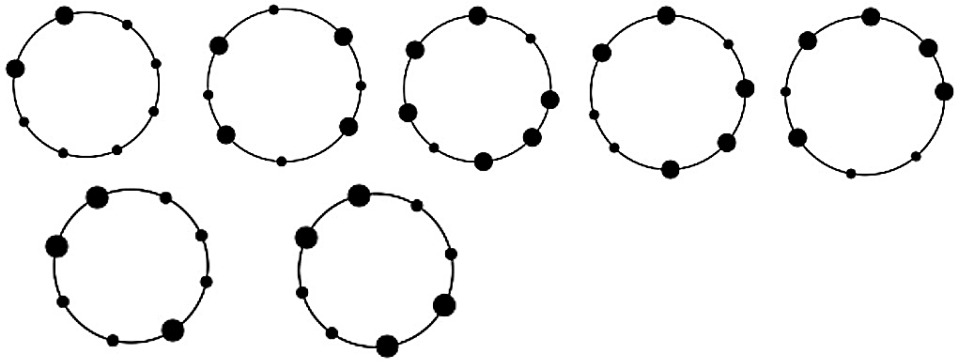


図９．n=8 のときの残存列

上述の n=8までの結果から，n が４の倍数のときに，多くの残存列が存在することが確認できた．しかし，n が非常に大きくなれば，残存列も多数存在することになるため，手作業では困難となる．そこで，Excel の VBA によって n が大きいときの残存列となる初期列を見つけることにした（Appendix 参照）．残存列となる初期列は，様々な経過を辿り最終的な残存列に収束する最終形についても調べた．次のシミュレーション結果は，n=8 のときの出力である．

表１．n=8 のときの残存列一覧

n=8 digits		
decimal number	binary number	final state
3	A_00000011	E_00000011
6	A_00000110	E_00000110
12	A_00001100	E_00001100

19	A_00010011	E_00000011
24	A_00011000	E_00011000
25	A_00011001	E_00011000
35	A_00100011	E_00000011
38	A_00100110	E_00000110
47	A_00101111	E_00011000
48	A_00110000	E_00110000
49	A_00110001	E_00110000
50	A_00110010	E_00110000
51	A_00110011	E_00110011
55	A_00110111	E_00000110
59	A_00111011	E_00011000
61	A_00111101	E_00000110
70	A_01000110	E_00000110
76	A_01001100	E_00001100
79	A_01001111	E_10000001
85	A_01010101	E_01010101
94	A_01011110	E_00110000
96	A_01100000	E_01100000
98	A_01100010	E_01100000
100	A_01100100	E_01100000
102	A_01100110	E_01100110
103	A_01100111	E_00000011
110	A_01101110	E_00001100
115	A_01110011	E_01100000
118	A_01110110	E_00110000
119	A_01110111	E_01110111
121	A_01111001	E_11000000
122	A_01111010	E_00001100
129	A_10000001	E_10000001
137	A_10001001	E_10000001
140	A_10001100	E_00001100
145	A_10010001	E_10000001
151	A_10010111	E_00001100
152	A_10011000	E_00011000
153	A_10011001	E_10011001
155	A_10011011	E_00000011
157	A_10011101	E_00001100
158	A_10011110	E_00000011
167	A_10100111	E_11000000

170	A_10101010	E_10101010
179	A_10110011	E_10000001
185	A_10111001	E_00110000
187	A_10111011	E_10111011
188	A_10111100	E_01100000
192	A_11000000	E_11000000
196	A_11000100	E_11000000
200	A_11001000	E_11000000
203	A_11001011	E_00000110
204	A_11001100	E_11001100
205	A_11001101	E_10000001
206	A_11001110	E_00000110
211	A_11010011	E_01100000
217	A_11011001	E_11000000
220	A_11011100	E_00011000
221	A_11011101	E_11011101
229	A_11100101	E_00000011
230	A_11100110	E_11000000
233	A_11101001	E_00110000
236	A_11101100	E_01100000
238	A_11101110	E_11101110
242	A_11110010	E_10000001
244	A_11110100	E_00011000

表１の第１列目は，８桁の２進数の10進法表示である．２列目は２進数表示であるが，各数字の前に A が付いているのは，８桁表示したときに，00000011の様に前に０を表示させるためのもので意味はない．３列目は，２列目の初期列の収束形を表示している．各数字の前に，E が付いているのは，２列目と同様の理由である．残存列は，66個出力されたが，円順列を考慮していないので，同じものを重複している．

我々は，n=20までの残存列を発見するシミュレーションを実行した．n=20のときには，結果を出力するためには，１時間以上の計算時間を要した．ここでは，その結果を掲載することは割愛するが，同じものを含んでいるとはいえ，膨大な554, 445個あることが分かった．よって，どのような初期列が残存列となるかを分類することは不可能であると判断した. そこで，残存列となる最終形に注目すれば次のような性質があることを発見した.

［Property Ⅰ］

初期列［ 1, 1, 0, 0, *, *, … *, 0, 0 ］は残存列となる．ただし，* は任意である．

また，セルの数 n が奇数のときは，残存列となる場合はこのパターンしかない．■

n が偶数の場合は，これ以外のパターンが存在する．［ 1, 0, 1, 0, 1, 0, …, 1, 0 ］は循環して残存列となる．また，n が４の倍数のときは，

［ 1, 1, 1, 0, 1, 1, 1, 0, …, 1, 1, 1, 0 ］は循環して残存列となる．

以上のことから次のようなことが予測される．

［ Conjecture ］

すべての残存列となる最終形は，次の３つのタイプでしかない．

タイプⅠ　［ 1, 1, 0, 0, 0, …, 0, 0, 1, 1, 0, 0, …, 0, 0 ］

タイプⅡ　［ 1, 0, 1, 0, 1, 0, …, 1, 0 ］

タイプⅢ　［ 1, 1, 1, 0, 1, 1, 1, 0, …, 1, 1, 1, 0 ］ ■

現時点でのこの予想の証明は完了していないが，n=20までのシミュレーションによってすべてを確認している．今後の課題は，任意の初期列はすべて３つのパターンに収斂することを数学的に証明することである．

第３章　２重円上のセル・オートマトンモデル

第２章の１重円におけるセル・オートマトンモデルを発展させ，図10に示すように２重円としたときには，どのような現象が見られるについて分析を行う．ルールは，第２章と同様であるが，上下のセルが隣接することが異なる．自分自身を含めて両隣，上下で，生存しているセルの数が２個のときは，次世代では生き残るとしたが，３個にすれば別なモデルができることになる．

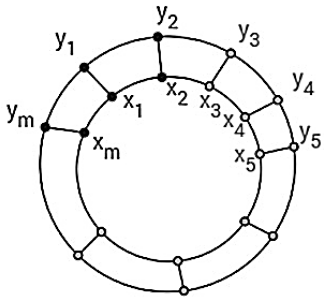


図 10.　２重円上のセル・オートマトンモデル

これらのモデルを数学的記述すると，以下のようになる．

初期列を $\begin{pmatrix} X_1(0), X_2(0), \cdots, X_n(0) \\ Y_1(0), Y_2(0), \cdots, Y_n(0) \end{pmatrix}$ と与える．

これらの初期列は，次の規則に従ってステップ（generation）を進める．

$$X_k(i+1)=\begin{cases} 1, & \text{if } X_{k-1}(i)+X_k(i)+X_{k+1}(i)+Y_k(i)=2 \\ & \qquad (\ k=2, 3, \cdots, n-1\ )(\ i=0, 1, 2, \cdots\ ) \\ 0, & \text{otherwise} \end{cases}$$

$$Y_k(i+1)=\begin{cases} 1, & \text{if } Y_{k-1}(i)+Y_k(i)+Y_{k+1}(i)+X_k(i)=2 \\ & \qquad (\ k=2, 3, \cdots, n-1\ )(\ i=0, 1, 2, \cdots\ ) \\ 0, & \text{otherwise} \end{cases}$$

ただし, k=1 のときは

$$X_1(i+1)=\begin{cases}1, & \text{if } X_n(i)+X_1(i)+X_2(i)+Y_1(i)=2\\ 0, & \text{otherwise}\end{cases} \quad (i=0, 1, 2, \cdots)$$

$$Y_1(i+1)=\begin{cases}1, & \text{if } Y_n(i)+Y_1(i)+Y_2(i)+X_1(i)=2\\ 0, & \text{otherwise}\end{cases} \quad (i=0, 1, 2, \cdots)$$

k=n のときは

$$X_n(i+1)=\begin{cases}1, & \text{if } X_{n-1}(i)+X_n(i)+X_1(i)+Y_n(i)=2\\ 0, & \text{otherwise}\end{cases} \quad (i=0, 1, 2, \cdots)$$

$$Y_n(i+1)=\begin{cases}1, & \text{if } Y_{n-1}(i)+Y_n(i)+Y_1(i)+X_n(i)=2\\ 0, & \text{otherwise}\end{cases} \quad (i=0, 1, 2, \cdots)$$

初期列 $\begin{pmatrix} X_1(0), X_2(0), \cdots, X_n(0) \\ Y_1(0), Y_2(0), \cdots, Y_n(0) \end{pmatrix}$ は，残存列または死滅列に分類できる．

１重円モデルにおいては，圧倒的に死滅列が多かったが，２重円モデルにおいては，残存列が多く存在する．一般的なパターンを見出すのは困難であるので，n=3のときに，１の数（残りはすべて０）によって分類したときの結果を記載する．

（ｉ）１が１個のとき
　すべての初期列は死滅列である．

（ii）１が２個のとき
　すべての初期列は残存列である．

（iii）１が３個のとき

$\begin{pmatrix} 1, 1, 1 \\ 0, 0, 0 \end{pmatrix}$ 以外はすべて残存列となる．

（iv）１が４個のとき

$\begin{pmatrix} 1, 1, 1 \\ 1, 0, 0 \end{pmatrix}$ 以外はすべて残存列となる．

（ｖ）１が５個，６個のとき
　すべての初期列は死滅列である．

これらの観察から，次のような性質が成立することがわかる．

［Property Ⅱ］

n ≧3のとき，１が１個しか含まない初期列は，すべて死滅列である．　■

［Property Ⅲ］

n ≧3のとき，１が２個しか含まない初期列が生存列となる場合は，２個の１が上下，左右，斜めに配置されたときだけである．ただし，n ＝4のときの

$\begin{pmatrix} 1, 0, 1, 0 \\ 0, 0, 0, 0 \end{pmatrix}$ は残存列となる例外が存在する．　■

［Property Ⅳ］

n ≧5のとき，１が３個しか含まない初期列が残存列となる場合は，

$\begin{pmatrix} 0, 0, 0, \cdots 1, 1, \cdots 0, 0, 0 \\ 0, 0, 0, \cdots 1, 0, \cdots 0, 0, 0 \end{pmatrix}$ のような配列のときだけである．

ただし，Property Ⅲの性質を持ち，次のように１個の１が影響を及ぼさないところにある場合を除く．

$\begin{pmatrix} 0, 0, 0, \cdots 0, 1, \cdots 0, 1, 0 \\ 0, 0, 0, \cdots 1, 0, \cdots 0, 0, 0 \end{pmatrix}$

また，n=6のときは，次のような例外がある．

$\begin{pmatrix} 1, 0, 1, 0, 1, 0 \\ 0, 0, 0, 0, 0, 0 \end{pmatrix}$

■

［Property Ⅴ］

これまでの［Property Ⅱ］［Property Ⅲ］［Property Ⅳ］に対して，１を０に置き換えても成立する双対性が成立する．　■

n=4の場合でこの双対性を確認してみる．

（ⅰ）１が１個だけの初期列はすべて死滅列である．

０が１個だけの初期列はすべて死滅列である．

（ⅱ）１が２個だけのとき，残存列となる初期列は以下の場合だけである．

$\begin{pmatrix} 1, 0, 0, 0 \\ 1, 0, 0, 0 \end{pmatrix}$　$\begin{pmatrix} 1, 0, 0, 0 \\ 0, 1, 0, 0 \end{pmatrix}$　$\begin{pmatrix} 1, 1, 0, 0 \\ 0, 0, 0, 0 \end{pmatrix}$　$\begin{pmatrix} 1, 0, 1, 0 \\ 0, 0, 0, 0 \end{pmatrix}$

０が２個だけのとき，残存列となる初期列は以下の場合だけである．

$\begin{pmatrix} 0, 1, 1, 1 \\ 0, 1, 1, 1 \end{pmatrix}$　$\begin{pmatrix} 0, 1, 1, 1 \\ 1, 0, 1, 1 \end{pmatrix}$　$\begin{pmatrix} 0, 0, 1, 1 \\ 1, 1, 1, 1 \end{pmatrix}$　$\begin{pmatrix} 0, 1, 0, 1 \\ 1, 1, 1, 1 \end{pmatrix}$

（iii）１が３個だけのとき，残存列となる初期列は以下の場合だけである．

$$\begin{pmatrix} 1,1,0,0 \\ 1,0,0,0 \end{pmatrix} \quad \begin{pmatrix} 1,0,1,0 \\ 0,1,0,0 \end{pmatrix} \quad \begin{pmatrix} 1,1,1,0 \\ 0,0,0,0 \end{pmatrix}$$

０が３個だけのとき，残存列となる初期列は以下の場合だけである．

$$\begin{pmatrix} 0,0,1,1 \\ 0,1,1,1 \end{pmatrix} \quad \begin{pmatrix} 0,1,0,1 \\ 1,0,1,1 \end{pmatrix} \quad \begin{pmatrix} 0,0,0,1 \\ 1,1,1,1 \end{pmatrix}$$

（iv）１が４個のときは，初期列が残存列となる場合は存在しない．

０が４個のときは，初期列が残存列となる場合は存在しない．

以上ですべての場合が尽くされたことになる．

第４章　考察と今後の課題

１重円におけるセル・オートマトンモデルでは，どのような初期列が残存列と成るかを調べた．一定のパターンは発見することができたが，すべてを統括する理論とすることができなかった．

シミュレーションにより，途中の世代では，高々４世代で定常状態に達することが分かっているが，証明には至っていない．しかし，最終的な収斂パターンが３タイプあることが予想できたことには意義がある．このモデルにおけるタイプⅠは，自分自身が生き残るためには，孤立した中で，２個のセルが隣り合って生存するしかないということを意味している．生物集団で考えれば，周りから孤立したところで，最小集団で生き延びることが可能であることを示唆している．情報や文化の伝播を考えれば，２人だけの情報や文化は維持できることを意味している．循環するタイプとして残り２つのタイプがあるが，１つのセルに注目すれば，生存と死滅を繰り返すことになる．すなわち，ヨーロッパの小さな国が，大きな国からの支配と独立を繰り返した歴史を重ねて見ることも可能である．現在のヨーロッパはEUとして統合された形式をとっているが，過密状態にあれば死滅するルールが適応されるなら，今後は大きく変化する可能性も含まれている．

閉じた社会の中での複雑な現象をモデル化するために，円周上にセルを配置したが，２重円にすることで新たな数学的な規則性を生み出すことが可能となった．円が２重となったことにより，状況は更に複雑さを増した．生存するセルの数を限定することによって，残存列となる性質を発見したが，一般的な法則を見つけ出すには至っていない．非常に興味深い現象は，１重円には無かった０と１の双対性がある．この現象が，このモデルを複雑にしている原因でもある．ある初期列が残存列となれば，その初期列の０と１を入れ替えても残存列となるようである．１重円では，孤立した中で最小の集団なら永遠に生き延びることが可能であったが，２重円では発見した残存列と全く正反対の過密集団を作れば，それは必ず孤立した元の集団に収斂して生存する現象が確認できている．これらの現象を数学的に証明できれば，大きな業績となる．本研究では，２つのモデルでのルールを共通としたが，これらのルールを変えれば新しいモデルとその現象を見ることができる

が，それは今後の課題としたい．

本論文では，これらモデルの具体的な現象への適応は行っていないが，今後の課題としてパンデミックを引き起こす細菌による伝染現象をモデルとしたシミュレーションを構築したい．現在のモデルでのルールは確定的に現象が推移するが，感染率などの確率現象を組み込むことで，新しい研究分野を開拓できるものと確信している．

［ Acknowledgements ］
The idea of cycle model with game of life is from Mr. Naoya Uematsu who attends the University of Northern Iowa. This is the part of the collaborative research with Koyu Uematsu. Authors appreciate Mr. Naoya Uematsu who checked the mathematical part of this paper and the English part of this abstract.

注

（1）Poundstone, W．有澤　誠訳，『ライフゲームの宇宙』新装版，日本評論社，2013年.
（2）von Neumann, J. “Theory of Self ―Reproducing Automata”, University of Illinois Press, 1966.
（3）Axelrod, R. “The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration”, Princeton, New Jersey: Princeton University Press, 1997.
（4）藤崎祥見，山野健太郎，秋山英三，「新文化の創造と伝播における伝統文化の影響と遠隔地相互作用の影響」，URL:mas.kke.co.jp/event/mas_competition5/result/10_paper.pdf.
（5）柳沢大地，西成活裕，「渋滞学のセルオートマトンモデル」，The Japan Society for Industrial and Applied Mathematics, Vol. 22, No. 1, pp. 2-14, 2012.

参考文献

マーチン・ガードナー，一松　信訳，『マーチン・ガードナーの数学ゲームⅠ』，日経サイエンス社，2010.
Packard, N.H. and S. Wolfram “Two-dimensional cellular automata”, J. Statistical Physics, 38(5/6), pp. 901–903, 1985.
Wolfram, S. “Universality and complexity in cellular automata” Rev. Mod. Phys., 55(3), pp. 1–4, 1983.

Appendix（VBA によるマクロ）

```
Sub 数値計算()
    ‘プログラム変数の定義
Dim i As Integer
Dim j As Integer
Dim k As Long
Dim count As Long
Dim シート名 As Variant
Dim 結果_Sht As Worksheet
Dim 桁数 As Integer
Dim 遷移回数 As Integer
Dim 元配列 As String
Dim 最終配列 As String
Dim 配列() As Integer
Dim 計算配列() As Integer
Dim 演算結果 As Integer
Dim 演算合計 As Integer

    '桁数と遷移回数の入力
桁数 = InputBox("生成する配列の桁数を入力してください.")
    '桁数 = 20
遷移回数 = InputBox("遷移する回数を入力してください.")
    '遷移回数 = 50
    '初期値
演算合計 = 0
count = 1
ReDim 配列(桁数 + 1)
ReDim 計算配列(桁数 + 1)
        '結果の書き出しSheetの生成
Worksheets.Add after:=Worksheets(Worksheets.count)
シート名 = "結果(" & 桁数 & "桁)" & Year(Date) & "." & Month(Date) & "." & Day
(Date) & "." & Hour(Time) & "." & Minute(Time) & "." & Second(Time)
ActiveSheet.Name = シート名
Set 結果_Sht = Worksheets(シート名)
Range("A1").Cells(1, 1) = "No."
Range("A1").Cells(1, 2) = "配列結果(" & 桁数 & "桁)"
        '===== シミュレーション開始 =====
```

```
For k = 0 To 2 ^ 桁数－1
        '配列の生成
    元配列 = DECtoBIN(桁数, k)
    最終配列 = ""
        '元配列から配列への転記
    配列(0) = Mid(元配列, 桁数, 1)
    For i = 1 To 桁数
        配列(i) = Mid(元配列, i, 1)
    Next i
    配列(桁数 + 1) = Mid(元配列, 1, 1)

    For j = 1 To 遷移回数
        演算合計 = 0
          '３つの配列の演算
        For i = 1 To 桁数
            演算結果 = 配列(i - 1) + 配列(i) + 配列(i + 1)
            If 演算結果 = 2 Then
                計算配列(i) = 1
            Else
                計算配列(i) = 0
            End If
            演算合計 = 演算合計 + 計算配列(i)
        Next i
        If 演算合計 = 0 Then
            Exit For
        End If
            '計算配列から配列への転記
        配列(0) = 計算配列(桁数)
        For i = 1 To 桁数
            配列(i) = 計算配列(i)
        Next i
        配列(桁数 + 1) = 計算配列(1)
    Next j
    For i = 1 To 桁数
        最終配列 = 最終配列 & 配列(i)
    Next i
    If 演算合計 <> 0 Then
    '     Range("A1").Cells(count + 1, 1).Activate
```

```
            Range("A1").Cells(count + 1, 1) = k
            Range("A1").Cells(count + 1, 2) = "A_" & 元配列
            Range("A1").Cells(count + 1, 3) = "E_" & 最終配列
            count = count + 1
        End If
    Next k
    End Sub

    Function DECtoBIN(桁数 As Integer, 数値 As Long) As String
    Dim myYard As Long
    Dim myNumber As Long
    Dim myExpo As Long
        myNumber = 数値
        While 2 ^ myYard <= myNumber
            myYard = myYard + 1
        Wend
        For i = 桁数 To myYard + 1 Step -1
            DECtoBIN = DECtoBIN & "0"
        Next i
        For myExpo = myYard - 1 To 0 Step -1
            If myNumber >= 2 ^ myExpo Then
                DECtoBIN = DECtoBIN & "1"
                myNumber = myNumber - 2 ^ myExpo
            Else
                DECtoBIN = DECtoBIN & "0"
            End If
        Next
    End Function
```