KIPS Trans. Softw. and Data Eng.
Vol.8, No.5 pp.187~192 pISSN: 2287-5905

https://doi.org/10.3745/KTSDE.2019.8.5.187

Finding State Transition Functions of One-Dimensional Cellular Automata by Evolutionary Algorithms

Jongwoo Park† · Sehee Wang†† · Kyubum Wee†††

ABSTRACT

Majority problem and synchronization problem on cellular automata(CA) are hard to solve, since they are global problems while CA operate on local information. This paper proposes a way to find state transition rules of these problems. The rules of CA are represented as CMR(conditionally matching rules) and evolutionary algorithms are applied to find rules. We find many solution rules to these problems, compared the results with the previous studies, and demonstrated the effectiveness of CMR on one-dimensional cellular automata.

Keywords : Cellular Automata, Majority Problem, Synchronization Problem, Evolutionary Algorithms, Conditionally Matching Rules

일차원 셀룰러 오토마타 상에서 진화 알고리즘을 이용한 상태전이함수 찾기

박 종 우† · 왕 세 희†† · 위 규 범†††

요 약

일차원 셀룰러 오토마타(CA)에서 과반수 문제(majority problem)와 동기화 문제(synchronization problem)는 국소 정보(local information)를 이용하여 전역 문제(global problem)를 풀어야 하는 계산적으로 어려운 문제이다. 본 논문에서는 일차원 CA에서 과반수 문제와 동기화 문제를 푸는 CA의 규칙을 찾는 방법을 제안한다. CA의 상태전이 함수(state transition function)를 일반적으로 사용하는 규칙표(rule table)가 아닌 조건부 매칭 규칙(CMR)으로 나타내고 진화 알고리즘을 적용하였다. 각 문제에서 다수의 규칙들을 찾아내어 제안한 방법을 효과적으로 사용할 수 있음을 보였다. 또한 이전 연구 결과와 비교하여 과반수 문제와 동기화 문제에서 CMR을 사용하는 방식의 효용성을 보였으며, 다른 일차원 CA 문제에도 CMR을 활용할 수 있는 가능성을 보였다.

키워드 : 셀룰러 오토마타, 과반수 문제, 동기화 문제, 진화 알고리즘, 조건부 매칭 규칙

1. 서 론

셀룰러 오토마타(cellular automata; CA)는 규칙인 격자 형태의 공간에서 유한한 상태(state)를 가지는 셀(cell)들로 정의된다. 각 셀에 대한 이웃(neighborhood)은 그 셀에 대한 관계로 정의한다. 시간 $t=0$ 일 때, 각 셀의 상태의 조합을 초기 구성(initial configuration; IC)이라 한다. 시간 $t-1$에서의 각 셀과 이웃들의 상태에 따라 상태전이함수(transition function)에 의해 시간 t에서의 각 셀의 상태가 정해진다. CA는 소수를 구하는 문제, 동기화 문제, 과반수 문제 등 계산적인 문제[1, 2], 자기 복제와 같은 생명체의 특성을 시뮬레이션하는 인공 생명 분야의 문제[3, 4], 또는 물리적 현상을 설명하는 모델[5] 등의 문제에 사용되고 있다.

과반수 문제(majority problem)와 동기화 문제(synchronization problem)는 이진 상태(binary state)를 가지는 일차원 CA상의 문제이다. CA를 진행시키는 시간 단계(time steps)의 수를 S라 하면 과반수 문제는 Fig. 1과 같이 S번의 시간 단계만큼 진행시켰을 때 모든 셀의 상태가 IC에서 과반수를 이루었던 상태로 수렴하도록 만드는 문제이다. 어떤 셀의 다음 상태를 결정할 때, 전체 셀의 상태를 인식하지 못하고 자신과 이웃 셀의 상태만으로 다음 상태가 결정되어 정확하게 풀기 어려운 문제 중 하나이다. 특히 0인 상태와 1인 상태의 수가 비슷한 경우에는 과반수를 이루는 상태를 찾기 어렵다. 동기화 문제는 어떤 IC로 시작하더라도 Fig. 2와 같이 CA를 S번의 시간 단계만큼 진행시켰을 때 모든 셀의 상태가 1인 경우와 모든 셀의 상태가 0인 경우가 반복하도록 만드는 문제이다.

일차원 CA에 관한 연구는 Wolfram[1]의 책에 방대한 내

※ 이 논문은 2018년도 한국정보처리학회 추계학술발표대회에서 '셀룰러 오토마타 상에서 과반수 문제의 상태전이 함수'의 제목으로 발표된 논문을 확장한 것임.
† 준 회 원 : 아주대학교 컴퓨터공학과 석사
†† 비 회 원 : 아주대학교 컴퓨터공학과 석·박사통합과정
††† 종신회원 : 아주대학교 소프트웨어학과 교수
Manuscript Received : December 17, 2018
First Revision : February 18, 2019
Accepted : March 1, 2019
* Corresponding Author : Kyubum Wee(kbwee@ajou.ac.kr)

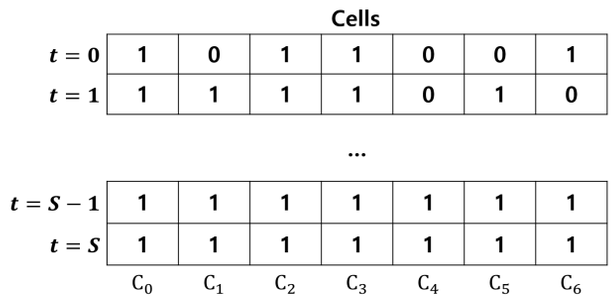

Fig. 1. An Illustration of Majority Problem

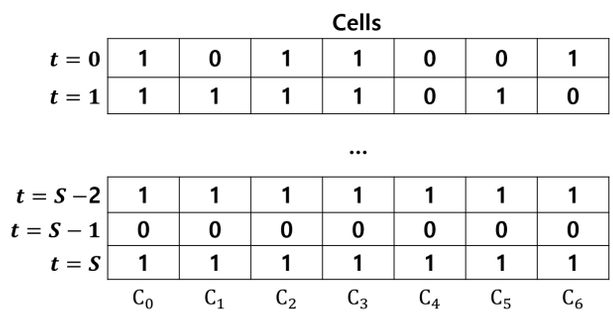

Fig. 2. An Illustration of Synchronization Problem

용이 잘 기술되어 있다. 과반수 문제와 동기화 문제는 Gacs-Kurdyumov-Levin[6]과 Mitchell[7]의 의해서 연구되었다. Gacs-Kurdyumov-Levin(GKL) 규칙은 일차원 공간에서 신뢰할 수 있는 계산과 상전이(phase transition)을 연구하기 위해 설계된 규칙으로 과반수 문제를 위해 설계된 규칙은 아니다. 그러나 과반수 문제에서 우수한 성능을 보이는 규칙으로 알려져 있다[6]. Mitchell은 셀과 그 이웃 셀들이 가질 수 있는 모든 경우의 상태에 대응하는 다음 상태 값을 표로 나타낸 규칙 표(rule table)를 전이 함수로 사용하고, 진화 알고리즘을 이용하여 동기화 문제와 과반수 문제의 규칙 표를 찾아내었다[7].

조건부 매칭 규칙(conditionally matching rules; CMR)은 Bidlo가 이차원 CA상에서 자기 복제 규칙을 찾기 위해 고안한 방법으로 상태전이함수를 나타내는 방법 중 하나이다. Bidlo는 CMR과 진화 알고리즘을 이용하여 이차원 CA상에서 자기 복제를 하는 규칙을 효과적으로 찾아낼 수 있음을 보였다[8-10].

본 연구에서는 CMR과 진화 알고리즘을 사용하는 것이 이차원 CA 만이 아니라 일차원 CA의 다양한 문제를 해결하는 데에도 효과적으로 사용될 수 있음을 보였으며, 과반수 문제와 동기화 문제에 적용하여 검증하였다.

본 논문의 제 2장에서는 CMR에 대해서 설명하고 제 3장에서는 CMR에 진화 알고리즘을 적용하는 방법에 대하여 설명한다. 제 4장에서는 CMR을 이용해 과반수 문제를 푸는 규칙을 찾기 위해 수행한 실험의 방법과 결과에 대해서 설명한다. 제 5장에서는 CMR을 이용해 동기화 문제를 푸는 규칙을 찾기 위해 수행한 실험의 방법과 실험 결과에 대해서 설명한다. 마지막으로 제 6장에서는 본 연구에 대한 결론에 대해서 설명한다.

2. 조건부 매칭 규칙

기존 CA의 전이 함수는 규칙 표가 사용되었다. 본 연구에서 사용한 CMR은 다수의 규칙(rule)으로 이루어져 있는데 각 규칙은 Fig. 3과 같이 셀과 각 이웃에 대한 조건 그리고 다음 상태 값으로 이루어져 있다. 본 연구에서 이웃은 좌우 각 r개의 셀로 정의한다. 각 조건은 비교 연산자와 비교할 값으로 이루어져 있다. 비교 연산자는 '같다', '같지 않다', '크거나 같다', '작거나 같다' 네 가지를 사용한다. 비교할 값과 다음 상태 값은 CA에서 사용할 수 있는 상태의 값들이 올 수 있다. 예를 들어 CA상에서 가질 수 있는 상태의 개수를 s 라고 했을 때, 0에서 $s-1$까지의 값을 가질 수 있다.

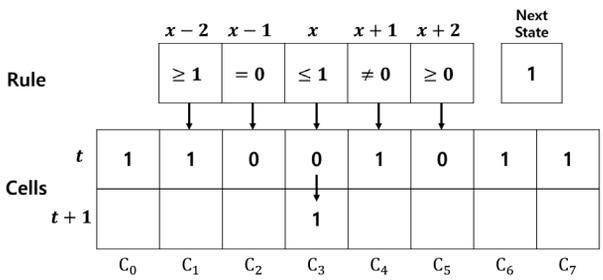

Fig. 3. An Illustration of State Change ($N=8,\ r=2$)

어떤 셀의 다음 상태를 결정할 때 CMR 규칙들의 조건을 순서대로 검사하고 먼저 모든 조건을 만족하는 규칙을 사용한다. 규칙의 개수는 실험 설정으로 정하고 진화 알고리즘 실행 중 규칙의 수는 변하지 않는다. 규칙의 조건을 검사하는 순서는 CMR이 처음 생성될 때 정해진다. 만약 규칙 중 조건을 만족하는 규칙이 존재하지 않는다면 현재 상태를 유지한다. 다음 상태를 결정할 셀의 위치를 x라 하면 어떤 규칙을 검사할 때 $x-r$부터 $x+r$까지 모든 조건을 만족하는지 검사한다. 일차원 CA 공간의 양 끝은 환형으로 연결된 구조이다. 즉 CA에서 셀의 개수를 N이라 하면 $N=8$인 CA $[C_0, C_1, \ldots C_7]$상에서 $C_{-1} = C_7$이다. 본 연구에서는 Mitchell이 찾아낸 규칙과 비교하기 위해 $r=3,\ N=149$로 설정하였다.

3. 진화 알고리즘

본 연구에서 사용한 CMR, CA, 진화 알고리즘의 다양한 연산자(opeator)와 매개변수(parameter)들은 이전 연구와 성능을 비교하기 위하여 Bidlo의 연구와 Mitchell의 연구에서 사용한 것과 같은 값을 사용하였다.

CMR의 집단(population)에 진화 알고리즘을 적용하여 다수결 문제와 동기화 문제의 해(solution)를 이루는 CMR을 찾아내었다. $r=3$이고 이진 상태를 가지는 CA에서 CMR 규칙의 수가 30인 경우 약 $4.78*10^{198}$ 가지의 CMR이 존재한다. 따라서 모든 경우의 CMR을 적용해 볼 수 없으므로 진화 알고리즘을 사용하여 해를 찾아내었다. 진화 알고리즘의 개체 수는 8로 설정하였으며, 첫 세대(generation)의 개체는 무작위

로 8개가 생성된다. 개체 수는 Bidlo가 제안한 방법의 설정을 따른 것이다. 개체 수가 많은 경우 한 세대를 계산하는 시간이 오래 걸리기 때문에 일반적인 진화 알고리즘보다 개체 수를 적게 설정하였다. 대신 충분한 세대 동안 진화시켰다.

개체의 복제와 돌연변이를 쉽게 하기 위해 CMR을 Fig. 4와 같이 정수 데이터로 표현한다. 규칙의 비교 연산자는 정수로 변환되며 '같다'는 0, '같지 않다'는 1, '크거나 같다'는 2, '작거나 같다'는 3으로 일대일 대응된다.

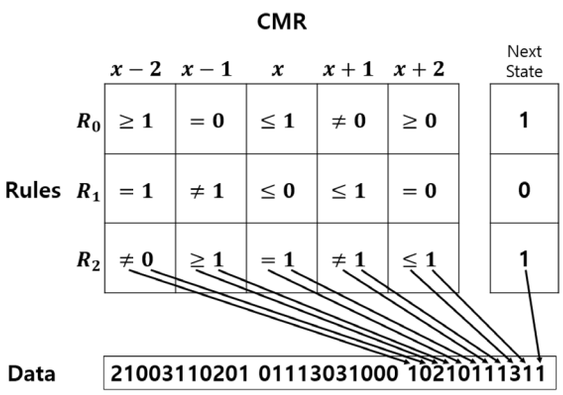

Fig. 4. How to Encode a Rule in CMR

진화 알고리즘은 다음 과정을 반복한다. (a) 개체를 이용하여 작업을 수행한다. (b) 적합도(fitness)를 계산한다. (c) 진화를 종료할지 결정한다. (d) 진화를 계속한다면 자식을 생성한다. (a)에서 (d)까지 한 번의 반복 실행은 한 세대 내에서 일어나고 다음 반복 실행에서는 이전 반복 실행의 (d) 과정에서 생성된 자식을 이용한다. 첫 세대부터 진화를 종료할 때까지를 진화 알고리즘을 한 번 실행(run) 했다고 한다.

CMR은 진화 알고리즘에서 하나의 개체를 이루고 각 개체는 서로 다른 다수의 IC에 대해서 CA를 수행한다. IC에서 상태가 0인 셀의 비율을 ρ라고 하자. 다수의 ρ를 정하여 각 ρ에 맞는 IC를 무작위로 생성한 뒤 CA를 수행한다. 각 IC는 $S=2N$ 만큼 진행시킨다.

하나의 개체를 다수의 IC에 적용하기 때문에 각 IC에 대해 적합도를 구하고 모든 IC에 대한 적합도의 합을 개체의 적합도로 정의한다. 적합도는 개체가 진화 목표에 얼마나 가까운지에 대한 값으로 진화 알고리즘에서 중요한 역할을 한다.

자식을 생성할 때 적합도가 더 높은 개체일수록 부모로 선택되어 자식으로 복제될 기회가 많아지기 때문에 적합도를 계산하는 방법에 따라 진화 과정이 달라진다. 따라서 문제에 따라 적합도 계산 방법이 다르고 적절한 계산 방법이 필요하다. 문제에 따른 적합도 계산 방법은 각 문제를 다루는 장에서 자세히 설명한다.

본 연구에서는 해당 세대에서 적합도가 목표값에 도달한 개체가 존재하거나 5,000 세대까지 진화를 반복하면 진화를 종료한다. 진화를 반복하는 최대 세대가 증가하면 적합도가 목표값에 도달할 가능성이 높아질 수 있으나 진화 알고리즘 실행에 더 많은 시간을 필요로 한다. 적합도의 목표값은 개체가 가질 수 있는 적합도의 최댓값이다.

(c)과정에서 진화를 종료하지 않고 계속하는 경우 다음 세대를 위해 자식을 생성해야 하는데 그 과정은 다음과 같다. (1) 무작위로 네 개의 개체를 선택한다. (2) 선택된 개체 중 가장 적합도가 높은 것을 부모로 하여 자식을 복제한다. (3) 자식에 돌연변이를 적용한다. (1)에서 (3)까지의 과정을 자식이 8개가 생성될 때까지 반복한다.

돌연변이는 개체의 진화를 위해 필요하며 다음과 같이 적용한다. 돌연변이를 일으킬 횟수를 정하기 위하여 0에서 2까지의 정수를 무작위로 생성한다. 생성된 정수가 0인 경우는 돌연변이를 일으키지 않는다. 그 외의 경우 개체의 정수 데이터에서 다른 값으로 교체할 정수를 선택한다. 선택된 정수가 CMR에서 무엇에 해당하는지에 따라 교체될 수 있는 정수가 다르다. 비교 연산자에 해당하는 정수가 선택되면 0에서 3까지의 정수 중 무작위로 선택된 값으로 교체하고, 비교할 값이나 다음 상태 값에 해당하는 정수가 선택되면 CA가 가질 수 있는 상태 중 하나로 교체한다.

진화 알고리즘 실행에서 찾아낸 개체가 CA 문제의 해를 얼마나 잘 구하는지 평가하기 위하여 성능을 측정하였다. 개체를 다수의 IC에 적용하여 해를 정확하게 구한 비율을 성능이라고 한다. 성능 측정을 위하여 IC를 생성하는 방법은 진화 알고리즘에서 개체의 적합도를 구하기 위하여 IC를 생성하는 방법으로 사용했던 ρ를 이용한다. $\rho \fallingdotseq 0.5$로 하여 과반수와 동기화를 이루는 상태를 찾기 어려운 IC를 10,000번 생성하여 개체를 적용한다.

4. 과반수 문제

4.1 적합도 계산

IC에서 CA를 S 단계만큼 진행시켜 N개의 셀 중에서 과반수 상태에 도달한 셀의 개수를 계산하고 이를 IC에 대한 적합도로 한다. 마지막 단계에서 최대한 많은 셀을 IC에서 과반수를 이루었던 상태로 만드는 개체가 부모로 선택되어 복제될 기회가 많아질 것이다. Fig. 5는 과반수문제의 적합도를 계산하는 예이다.

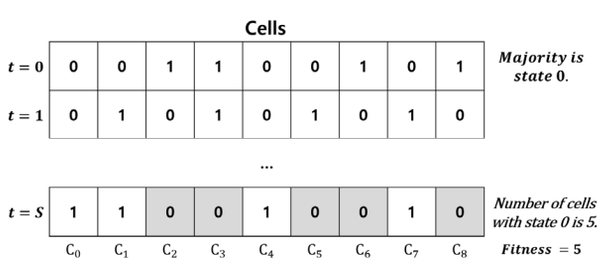

Fig. 5. How to Compute Fitness in Majority Problem

4.2 진화 알고리즘에서 IC 생성

개체의 적합도를 구하기 위하여 개체를 적용하는 IC를 생성하는 적절한 방법을 먼저 찾아야 한다. 따라서 ρ를 세 가지 방법으로 설정하여 IC를 생성해 실험하였다. IC를 생성하

는 각 방법을 A, B, C로 표기한다. A는 $\rho \fallingdotseq 0$에서 $\rho \fallingdotseq 1$까지 0.05의 간격으로 IC를 한 번씩 생성한다. B는 $\rho \fallingdotseq 0.5$로 IC를 20번 생성한다. N은 홀수이기 때문에 정확히 $\rho = 0.5$가 되지 않는다. IC를 생성할 때 각 셀의 상태를 1/2 확률로 정하여 0 또는 1로 설정한다. C는 $\rho \fallingdotseq 0.4$에서 $\rho \fallingdotseq 0.6$까지 0.01의 간격으로 한 번씩 생성한다.

개체의 적합도를 구하기 위한 IC를 생성하는 방법에 따라 CMR 규칙의 수를 30개로 설정하여 진화 알고리즘을 20회씩 실행하였다. 실험의 결과 Table 1은 각 방법의 진화 알고리즘 실행에서 최대 적합도에 도달한 횟수와 각 실험에서 찾은 개체들의 성능 중 가장 높은 수치를 나타내었다.

A는 진화 알고리즘의 모든 실행에서 최대 적합도에 도달한 개체를 찾지만, 최고 성능이 C보다 약 22% 떨어진다. 0인 상태가 확실히 많거나 1인 상태가 확실히 많은 경우 과반수를 이루는 상태를 구하기가 쉽기 때문에 성능이 낮은 개체도 최대 적합도에 도달하기 쉽다. 과반수 상태를 구하기 어려운 IC에서 해를 잘 구하지 못하는 결과를 보인다.

B는 과반수를 이루는 상태를 구하기 어려운 IC만 생성하여 CA를 수행하였다. 진화 알고리즘 실행에서 최대 적합도에 도달한 개체를 찾지 못하였다.

C는 두 번 중 한 번은 최대 적합도에 도달한 개체를 찾을 수 있었고, 실험에서 찾아낸 개체는 비교적 높은 성능을 보였다. 따라서 적합도를 계산하기 위한 IC를 생성하는 방법은 C에서 사용한 방법이 가장 적절하다고 할 수 있다.

Table 1. Result of 20 Runs by IC Sets

IC set	Number of runs max fitness reached	Performance of best CMR
A	20	51.79%
B	0	n. a.
C	11	73.35%

4.3 실험 결과

1) 진화 알고리즘 실행 결과

CMR 규칙의 수에 따라서 성능에 미치는 영향을 알아보기 위하여 CMR 규칙의 수를 30, 40, 50개로 설정하여 진화 알고리즘을 20회씩 실행하였다. 진화 알고리즘에서 IC를 생성하는 방법은 C와 동일하다.

Table 2. Result of 20 Runs by CMR Sizes

CMR size	Number of runs max fitness reached	Performance of best CMR	Average number of steps until majority is obtained (Best CMR)
CMR 30	11	73.35%	121
CMR 40	12	75.83%	74
CMR 50	12	75.13%	82
Mitchell	n. a.	76.90%	n. a.
GKL	n. a.	81.60%	n. a.

실험의 결과 Table 2에서 CMR 규칙의 수에 따라서 최대 적합도에 도달하는 개체를 찾아내는 수는 유의미한 차이를 보이지 않는다. CMR의 규칙이 40개일 때 다른 두 개의 실험보다 높은 성능을 보이지만 CMR의 규칙이 50개일 때와 큰 차이를 보이지 않는다. 또한 본 연구에서 찾아낸 개체는 Mitchell 이 찾아낸 규칙의 성능과 근접하여 CMR을 이용한 상태전이 방법이 과반수 문제의 규칙을 찾는데 효용성이 있음을 보여준다.

2) 찾아낸 규칙의 CA 실행 결과

(Fig. 6)은 본 연구의 실험에서 찾은 개체들 중 성능이 가장 뛰어난 것을 ρ값에 따라 실행한 것을 시각화 한 것이다. CA는 $2N = 298$단계만큼 수행했으나, N단계 이전에 이미 과반수 상태로 수렴하였으므로 $N = 149$ 단계까지만 보여준다. (Fig. 6A)는 $\rho \fallingdotseq 0.25$일 때의 실행 결과로 모든 셀이 0으로 수렴하는 것을 볼 수 있다. (Fig. 6B)는 $\rho \fallingdotseq 0.75$인 경우의 실행 결과로 모든 셀이 1로 수렴한다. (Fig. 6C), (Fig. 6D) 는 $\rho \fallingdotseq 0.5$로 설정하여 각각 0이 많은 경우와 1이 많은 경우의 실행 결과이며 과반수 상태를 정확히 구하는 과정을 보여준다.

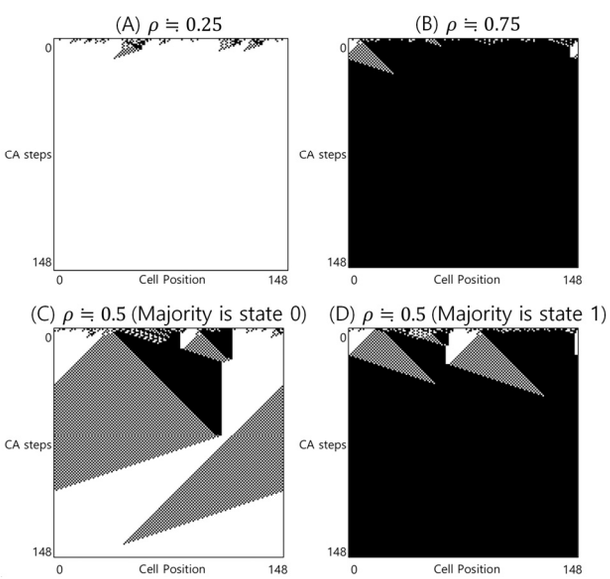

Fig. 6. Progression of Configurations Depending on ρ

5. 동기화 문제

5.1 적합도 계산

동기화 문제의 해를 정확하게 구하는지 검사하기 위해서 모든 셀의 상태가 1일 때 모두 0으로 전이 되는지 확인하고 모든 셀의 상태가 0일 때 모두 1로 전이되는지 확인해야 하므로 적어도 세 단계를 검사해야 한다. 따라서 IC에서 CA를 S시간 단계만큼 진행시켜 $S-2$부터 S 단계의 셀들로 적합도를 계산한다.

각 시간 단계에서 적합도를 계산하는 방법은 다음과 같다. $S-2$단계에서는 과반수를 이루는 상태 m(0 또는 1)을 찾고

상태 m의 개수를 계산한다. $S-1$단계에서는 $S-2$단계에서 상태 m인 셀 중 상태가 변한 셀의 개수를 계산한다. S단계에서는 $S-1$단계에서 m이 아닌 상태의 셀 중 상태가 변한 셀의 개수를 계산한다. 각 시간 단계에서 계산한 값의 합이 IC에 대한 적합도가 된다. 어떤 상태가 확실하게 많도록 만들고, 0인 상태는 1인 상태로 1인 상태는 0인 상태로 많이 만들수록 복제될 기회가 많아질 것이다. (Fig. 7)는 동기화 문제의 적합도를 구하는 예이다.

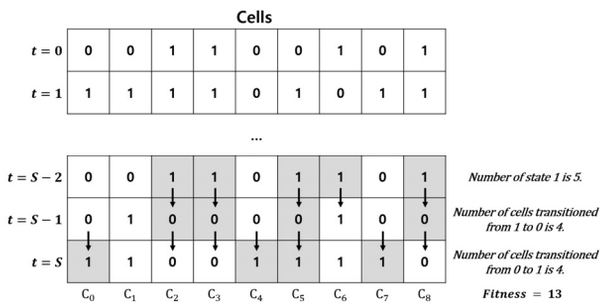

Fig. 7. How to Compute Fitness in Synchronization Problem

5.2 진화 알고리즘에서 IC 생성

동기화 문제는 과반수 문제와 마찬가지로 어떤 시간 t에서 모든 셀의 상태가 같아야 한다. 따라서 0인 상태의 수가 확실히 많거나 1인 상태의 수가 확실히 많을 때 모든 셀의 상태를 한 가지 상태로 만들기 쉽다. 그러므로 동기화 문제에서 IC를 생성하는 방법은 과반수 문제에서 가장 결과가 좋았던 $\rho \fallingdotseq 0.4$부터 $\rho \fallingdotseq 0.6$까지 0.01의 간격으로 IC를 1회씩 생성하는 방법을 사용하였다.

5.3 실험 결과

1) 진화 알고리즘 실행 결과

동기화 문제는 Mitchell이 찾아낸 규칙을 공개하여 직접 실행해 볼 수 있었다. 각 실험에서 찾아낸 가장 높은 성능의 CMR과 Mitchell 이 찾아낸 규칙을 $\rho \fallingdotseq 0.5$로 하여 IC를 10,000번 생성해 CA를 수행하였고 동기화 상태에 도달할 때까지의 단계수의 평균을 구하였다.

CMR 규칙의 수에 따라 진화 알고리즘을 20회씩 실행하였다. 실험의 결과 Table 3에서 CMR 규칙의 수가 40개 일 때 최대 적합도 도달 횟수가 진화 알고리즘 실행 20번 중 19번으로 가장 많았다. 규칙의 수가 30개일 때와 50개 일 때 각각 13번 12번 최대 적합도에 도달하였다, CMR 규칙의 수가 증가할수록 최고 성능이 증가하는 경향을 보였다. CMR 규칙의 수가 30개일 땐 복잡한 규칙을 잘 표현하지 못하여 규칙의 수가 40개일 때 보다 최대 적합도에 도달하는 횟수가 적은 것으로 보인다. CMR 규칙의 수가 50개일 때 복잡한 규칙을 잘 표현하여 뛰어난 성능을 보이는 개체를 찾을 수 있었으나 그만큼 규칙은 찾기 어려워져 최대 적합도에 도달하는 횟수가 적은 것으로 추측된다.

가장 좋은 성능을 보이는 개체는 100%에 조금 못 미치는 것을 볼 수 있다. 그러나 CMR 규칙의 수가 40개와 50개에서 가장 좋은 성능을 보인 개체는 Mitchell의 규칙 보다 동기화 상태를 빠르게 구하였다. 동기화 상태에 도달하는 속도와 성능은 큰 상관관계가 없음을 알 수 있다.

Table 3. How Fast Synchronization is Reached

CMR size	Number of runs max fitness reached	Performance of best CMR	Average number of steps until synchronization (Best CMR)
CMR 30	13	95.59%	133
CMR 40	19	96.32%	25
CMR 50	12	99.38%	59
Mitchell	n. a.	100%	87

2) 찾아낸 규칙의 CA 실행 결과

Fig. 8은 CMR 규칙의 수에 따른 진화 알고리즘 실행에서 찾아낸 성능이 가장 좋은 개체들과 Mitchell의 규칙을 동일한 IC로 CA를 실행한 것을 시각화 한 것이다. CA는 $2N=298$ 단계만큼 수행했으나, N단계 이전에 이미 동기화 상태에 도달하였으므로 $N=149$ 단계까지만 보여준다. 각 규칙이 서로 다른 방법으로 동기화 상태에 도달하는 것을 볼 수 있다.

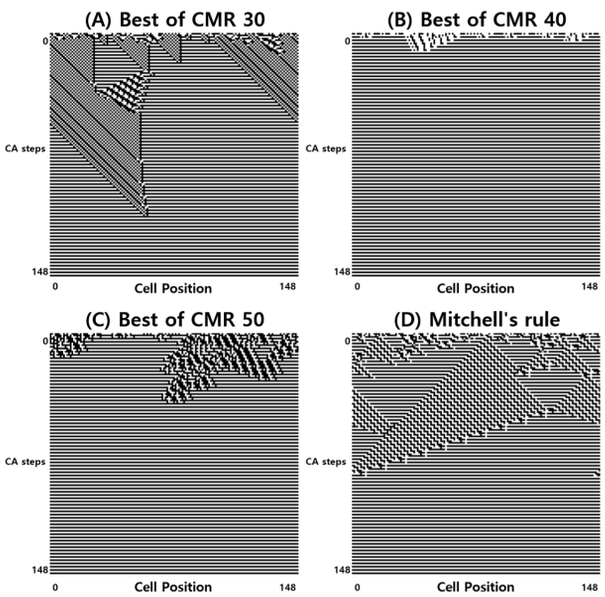

Fig. 8. Progression of Configurations Depending on Rules

6. 결 론

본 연구에서는 CA 상에서 많이 연구되고 잘 알려진 과반수 문제와 동기화 문제에 CMR을 적용하는 방법을 제안하였다. 제시한 방법으로 실험한 결과 과반수 문제와 동기화 문제를 푸는 다수의 CA를 찾아내어 CMR과 진화 알고리즘을 적용하는 것이 효과적인 방법임을 검증하였다. 과반수 문제의 진화 알고리즘 실험에서는 찾아낸 CA가 기존의 연구에서 과반수 문제의 CA와 유사한 성능을 보여주었다. 동기화 문제의

진화 알고리즘 실험에서 찾아낸 CA는 기존 연구의 CA보다 빠르게 동기화 상태를 만드는 비율이 높았다.

과반수 문제에서도 기존 연구의 결과보다 더 빠르게 과반수 상태에 도달할 것으로 기대한다. 기존 Mitchell의 연구에서는 과반수 문제의 가장 우수한 규칙을 공개하지 않았으므로, 얼마나 빠르게 과반수 상태에 도달하는지 계산할 수 없었다.

Bidlo는 CMR을 이차원 CA의 복제 문제에 적용하여 효용성을 보였으며, 일차원 CA상에서의 CMR의 효용성은 충분히 검증하지 않았다. 본 연구에서는 일차원 CA의 상태전이함수를 구하는 방법으로서 CMR과 진화알고리즘을 적용하는 것이 효과적임을 보였다.

IC를 생성하는 방법에 따른 실험과 CMR의 개수에 따른 실험으로 이들이 최대 적합도에 도달하는 횟수와 성능에 영향을 준다는 사실을 알 수 있었다. 진화 알고리즘에서 IC를 생성하는 방법과 CMR의 개수를 조절하고 진화 알고리즘의 실행 횟수를 증가시키면 본 연구에서 찾아낸 CA 보다 좋은 성능의 CA를 찾아낼 수 있을 것으로 생각된다.

적합도 함수를 개선하여 CA의 성능을 높일 수 있을 것으로 기대하여 연구를 진행 중이다. 또한 CMR은 이진 상태 CA보다 다수의 상태를 가지는 CA의 규칙을 표현하는데 적합하므로 다수의 상태를 가지는 과반수 문제와 동기화 문제로 확장하는 연구를 진행 중이다.

References

[1] S. Wolfram, “A new kind of science: A 15-year view,” *Complex Syst.*, Vol.26 No.5, pp.197-224, 2017.

[2] R. Das, J. P. Crutchfield, M. Mitchell and J. E. Hanson, “Evolving globally synchronized cellular automata,” in Proceedings of the 6th International Conference on Genetic Algorithms, 1995, pp.336-343.

[3] J. Von Neumann, “Theory of Self-Reproducing Automata,” A. W. Burks, Ed. Urbana, IL, USA: Univ. Illinois Press, 1966.

[4] C. G. Langton, “Studying artificial life with cellular automata,” *Phys D Nonlinear Phenom*, Vol.22, No.(1-3), pp.120-149, 1986.

[5] T. Ostoma and M. Trushyk, “Cellular automata theory and physics,” Arxiv Preprint physics/9907013, 1999.

[6] P. Gacs, G. L. Kurdyumov, and L. A. Levin, “One-dimensional uniform arrays that wash out finite islands,” *Problemy Peredachi Informatsii*, Vol.14, No.3, pp.92-96, 1978.

[7] M. Mitchell, “An Introduction to Genetic Algorithms,” MIT press, 1998.

[8] M. Bidlo, “On routine evolution of complex cellular automata,” *IEEE Transactions on Evolutionary Computation*, Vol.20 No.5, pp.742-754, 2016.

[9] M. Bidlo, “Evolving multiplication as emergent behavior in cellular automata using conditionally matching rules,” in *IEEE Congress on Evolutionary Computation (CEC)*, 2014, pp.2732-2739.

[10] M. Bidlo and Z. Vasicek, “Evolution of cellular automata with conditionally matching rules,” in *IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp.1178-1185.

박 종 우

https://orcid.org/0000-0001-9793-9978

e-mail : jwmtp2@ajou.ac.kr

2017년 동서대학교 컴퓨터공학과(학사)

2019년 아주대학교 컴퓨터공학과(석사)

관심분야 : 알고리즘

왕 세 희

https://orcid.org/0000-0001-9192-2411

e-mail : wsh0509@ajou.ac.kr

2015년 아주대학교 정보컴퓨터공학부(학사)

2015년～현 재 아주대학교 컴퓨터공학과 석・박사통합과정

관심분야 : 생물정보학, 머신러닝

위 규 범

https://orcid.org/0000-0003-1379-6995

e-mail : kbwee@ajou.ac.kr

1978년 서울대학교 수학과(학사)

1985년 University of Wisconsin 전산학과(석사)

1992년 Indiana University 전산학과(박사)

1993년～현 재 아주대학교 소프트웨어학과 교수

관심분야 : 컴퓨팅이론