Article:
Kim, J. orcid.org/0000-0002-3456-6614, Lee, W. and Cho, K.-H. (2024) Recursive Self-Composite Approach Towards Structural Understanding of Boolean Networks. IEEE/ACM Transactions on Computational Biology and Bioinformatics. ISSN 1545-5963

https://doi.org/10.1109/tcbb.2024.3415352

Recursive Self-Composite Approach Towards Structural Understanding of Boolean Networks

Jongrae Kim, *Senior Member, IEEE,* Woojeong Lee, and Kwang-Hyun Cho, *Senior Member, IEEE,*

***Abstract*—Boolean networks have been widely used in systems biology to study the dynamical characteristics of biological networks such as steady-states or cycles, yet there has been little attention to the dynamic properties of network structures. Here, we systematically reveal the core network structures using a recursive self-composite of the logic update rules. We find that all Boolean update rules exhibit repeated cyclic logic structures, where each converged logic leads to the same states, defined as kernel states. Consequently, the period of state cycles is upper bounded by the number of logics in the converged logic cycle. In order to uncover the underlying dynamical characteristics by exploiting the repeating structures, we propose leaping and filling algorithms. The algorithms provide a way to avoid large string explosions during the self-composition procedures. Finally, we present three examples—a simple network with a long feedback structure, a T-cell receptor network and a cancer network—to demonstrate the usefulness of the proposed algorithm.**

***Index Terms*—Boolean networks, logic structures, kernel states, biological networks, systems biology**

I. Introduction

BOOLEAN network formalism is a useful mathematical modelling approach to describe complex interactions and dynamics of biological systems [1]. In this formalism, individual biological entities, such as genes, proteins, or other molecular components, are represented by nodes, while their interactions are depicted as edges. These nodes are assigned with time-varying binary states – either on (active) or off (inactive) – thus facilitating a simplified modelling process and allowing for a broader range of interactions while still capturing essential dynamical properties. Logical relationships among these nodes are specified through Boolean functions. Following these rules, node states are updated synchronously or asynchronously, eventually converging to a stable state known as an attractor. It has been previously demonstrated that stable attractor states in gene regulatory networks correspond to distinct cellular phenotypes or cell fates [2]–[6]. In this regard, extensive studies have been done to investigate the long-term behaviour of biological networks represented by Boolean network models, aiming to predict real intra-cellular dynamics across various biological processes, including the cell cycle [7], differentiation [8], [9], and tumorigenesis [10], [11]. Such studies have not only enhanced our understanding of biological phenomena but also enabled the prediction of drug responses for precision medicine of complex diseases [12] and the identification of potential therapeutic targets for drug discovery [13], [14].

Manuscript received 30 January 2024. Revised 14 May 2024. This work was supported by the Cheney Fellow program of the University of Leeds, Leeds, UK. It was also supported by the National Research Foundation of Korea (NRF) grants funded by the Korea Government, the Ministry of Science and ICT [2023R1A2C3002619 and 2021M3A9I4024447 (Bio & Medical Technology Development Program)] and the internal fund/grant of Electronics and Telecommunications Research Institute (ETRI) [23RB1100, Exploratory and Strategic Research of ETRI-KAIST ICT Future Technology].

J. Kim is with the School of Mechanical Engineering, University of Leeds, Leeds LS2 9JT, UK (e-mail: menjkim@leeds.ac.uk).

W. Lee is with the Department of Bio and Brain Engineering, KAIST, Daejeon, Republic of Korea (e-mail: frship35@kaist.ac.kr).

K.-H. Cho is with the Department of Bio and Brain Engineering, KAIST, Daejeon, Republic of Korea (e-mail: ckh@kaist.ac.kr, Corresponding author).

Let us consider the Boolean networks given by

$$\begin{aligned} x_1(k+1) &= f_1[x_1(k), x_2(k), \ldots, x_{n-1}(k), x_n(k)] \\ x_2(k+1) &= f_2[x_1(k), x_2(k), \ldots, x_{n-1}(k), x_n(k)] \\ &\vdots \\ x_n(k+1) &= f_n[x_1(k), x_2(k), \ldots, x_{n-1}(k), x_n(k)] \end{aligned} \tag{1}$$

where $x_i(k)$ is the i-th Boolean state equal to either *true* (equivalently T or 1) or *false* (equivalently F or 0) at k for $i = 1, 2, \ldots, n-1, n$, k is the non-negative integer in $[0, \infty)$, $x_i(0)$ is the initial state, $f_i(\cdot)$ is a synchronous update rule consisting of the Boolean operations conjunction (and, $\wedge$), disjunction (or, $\vee$), and negation (not, $\neg$) and $x_i(k+1)$ is the updated Boolean state for $i = 1, 2, \ldots, n-1, n$.

The Boolean network shown in (1) can be written in a compact form as follows:

$$\mathbf{x}(k+1) = \mathbf{f}[\mathbf{x}(k)] \tag{2}$$

where

$$\mathbf{x}(k) = \begin{bmatrix} x_1(k) & x_2(k) & \ldots & x_n(k) \end{bmatrix}^T, \tag{3a}$$

$$\mathbf{f}[\mathbf{x}(k)] = \begin{bmatrix} f_1[\mathbf{x}(k)] & f_2[\mathbf{x}(k)] & \ldots & f_n[\mathbf{x}(k)] \end{bmatrix}^T \tag{3b}$$

and $(\cdot)^T$ is the transpose.

The state space is 2^n-dimensional and the main interest in Boolean network analysis is finding steady-states and periodic cycles. In the synchronous update of Boolean networks, every initial state converges to a steady state or a periodic cycle. As n increases, the dimension of the state space, 2^n, increases exponentially. Therefore, executing the exhaustive search to find attractors is infeasible even for moderate-size networks, e.g., n around 30. One of the well-known approaches in Boolean networks called the semi-tensor approach is also an exhaustive method [15]. The aggregation algorithm proposed in [16] relies on the specific modular structure of the networks. Hence, Boolean network analysis results are often obtained from probabilistic approaches based on simulations over a finite number of random samples. Finding attractors or control

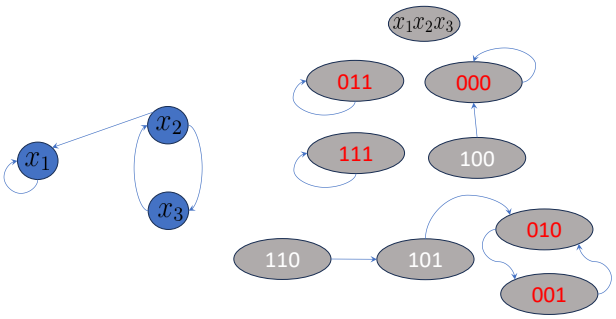

Fig. 1. The interaction graph and state transition map of the Boolean network, (4), are shown. There are three steady-states, $\{s = (000), (011), (111)\}$ and one cycle with the period 2, $\{s = (010) \leftrightarrow (001)\}$, where $s = (x_1x_2x_3)$.

strategies for Boolean networks is also known to be NP-hard [17].

On the other hand, there are rich theoretical results in the continuous system described by ordinary differential equations: $dx/dt = f(x)$, where $d(\cdot)/dt$ is the derivative with respect to time, t, and $f(x)$ is a nonlinear function satisfying the existence and uniqueness conditions of the solution. The equilibrium points and their stability are inherent properties of the right-hand side of the differential equation, i.e., $f(x)$. The solution of $f(x) = 0$ is the equilibrium point and the eigenvalues of $df(x)/dx$ at the equilibrium point provide the stability condition. The main motivation of our approach to be shown is from the question about whether the right-hand side of (2), i.e., $\mathbf{f}[\mathbf{x}(k)]$, can also provide any clue for the characteristics of Boolean networks. A similar motivation in [18] and [19] focusing on the network structures leads to the design of control of Boolean networks that is potentially applicable to a class of large-scale Boolean networks with strong structural controllability and stabilizability properties.

In the following sections, first, the motivation of the proposed method is illustrated with a simple toy example. Secondly, we present the main results of the recursive self-composition approach to investigate the structure of Boolean networks. Thirdly, we apply the proposed method to various examples including a simple network with a long feedback path and two biological networks – a T-cell signalling pathway and a cancer signalling network – highlighting the advantages of the proposed method. Finally, the conclusions are made.

II. Recursive Self-Composite Boolean Network

A. Motivations

Let us consider the following Boolean network model:

$$x_1(k+1) = x_1(k) \wedge x_2(k) \tag{4a}$$
$$x_2(k+1) = x_3(k) \tag{4b}$$
$$x_3(k+1) = x_2(k) \tag{4c}$$

The model has feedback loops connecting all three states and such feedback loops are important structural characteristics of many biological systems. Since the right-hand side of $x_1(k+1)$ in (4a) is of the conjunction of $x_1(k)$ and $x_2(k)$, the 75% of $x_1(k+1)$ is 0 (False), i.e., the 25% of $x_1(k+1)$ is 1 (True). All four possible outputs from the conjunction of $x_1(k)$ and $x_2(k)$ produce 0 except when both are 1. Using the same approach, examining the right-hand sides of $x_2(k+1)$ and $x_3(k+1)$ in (4), the probability that the output of $x_2(k+1)$ or $x_3(k+1)$ is 0 or 1 is 50%.

The question is how accurate these probabilities are with respect to the final state. The final state is a steady state or a state belonging to a cycle. Figure 1 shows the interaction graph and transition map, where the state, s, is equal to $(x_1x_2x_3)$. Three steady states and one cycle are shown in red. If we consider all eight states in Figure 1 and count the number of states with the final state $x_1 = 1$, there is only one case. In all other cases except $(x_1x_2x_3) = (111)$, x_1 becomes 0. Hence, the probability for x_1 equal to 0 in the final state is 87.5%, 7 out of 8. It is not equal to the 75% that was estimated earlier.

The cause for this difference is the usage of one-step propagation equation. So, a longer propagation would provide a better estimation. Any exact calculation by exhaustive numerical simulation considering all possible states is not feasible for large-size networks. Instead, let us consider the two-step propagation symbolically as follows:

$$\begin{aligned} x_1(k+2) &= x_1(k+1) \wedge x_2(k+1) \\ &= [x_1(k) \wedge x_2(k)] \wedge x_3(k) \\ &= x_1(k) \wedge x_2(k) \wedge x_3(k) \end{aligned} \tag{5a}$$
$$x_2(k+2) = x_3(k+1) = x_2(k) \tag{5b}$$
$$x_3(k+2) = x_2(k+1) = x_3(k) \tag{5c}$$

This two-step prediction is obtained by substituting the one-step prediction twice. Similarly, the three-step propagation is obtained as follows:

$$\begin{aligned} x_1(k+3) &= x_1(k+1) \wedge x_2(k+1) \wedge x_3(k+1) \\ &= [x_1(k) \wedge x_2(k)] \wedge x_3(k) \wedge x_2(k) \\ &= x_1(k) \wedge x_2(k) \wedge x_3(k) \end{aligned} \tag{6a}$$
$$x_2(k+3) = x_2(k+1) = x_3(k) \tag{6b}$$
$$x_3(k+3) = x_3(k+1) = x_2(k) \tag{6c}$$

The four-step propagation is given by

$$\begin{aligned} x_1(k+4) &= x_1(k+1) \wedge x_2(k+1) \wedge x_3(k+1) \\ &= x_1(k) \wedge x_2(k) \wedge x_3(k) \end{aligned} \tag{7a}$$
$$x_2(k+4) = x_3(k+1) = x_2(k) \tag{7b}$$
$$x_3(k+4) = x_2(k+1) = x_3(k) \tag{7c}$$

and the procedures find that the four-step propagation is the same as the two-step propagation. We refer to these substitution steps as the *recursive self-composition* procedure.

As shown in Figure 2, the update logic itself switches between the two update rules. While the x_2 and x_3 propagation rules cross-update between the two, the x_1 update rule converges to the conjunction of the three states. By inspecting the right-hand side of the converged update rule, the probability of x_1 converging to the final state equal to 0 is 7 out of 8, which coincides with the true probability.

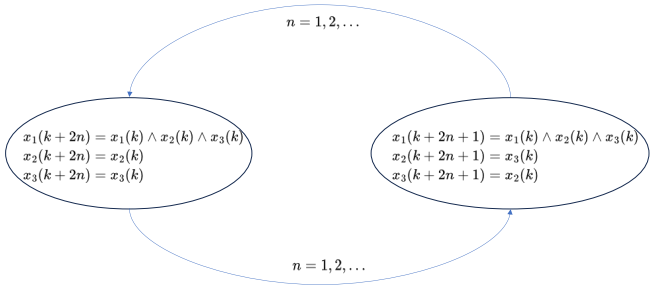

Fig. 2. The Boolean logic in (4) switches between two update rules.

In the exhaustive approach, the transition matrix, L, describes the updates of eight states in Figure 1 as follows:

$$\mathbf{s}_{k+1} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} (000) \\ (001) \\ (010) \\ (011) \\ (100) \\ (101) \\ (110) \\ (111) \end{bmatrix}}_{\mathbf{s}_k} \tag{8}$$

In the semi-tensor approach [15], $\mathbf{s}_k$ or $\mathbf{s}_{k+1}$ is the state vector with the element corresponding to the current state equal to 1 and the rest set to 0, i.e.,

$$\mathbf{s}_k = \begin{bmatrix} s_0 & s_1 & \ldots & s_{2^n-1} \end{bmatrix}^T \tag{9}$$

where $s_i = 1$ for i equal to the decimal number whose binary number corresponds to the current state $(x_1 x_2 \ldots x_n)$ and $s_i = 0$ for the others, where $i \in \{0, 1, 2, \ldots, 2^n - 1\}$. For instance, if the initial state for (x_1, x_2, x_3) is equal to (010), then, $s_2 = 1$ and the rest of s_i equal to 0.

$$\mathbf{s}_0 = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{10}$$

(010), i.e., $s_2 = 1$, is converted to (001), i.e., $s_1 = 1$, as shown in Figure 1, and $L\mathbf{s}_0$ provides the corresponding transition state, $\mathbf{s}_1$, i.e.,

$$\mathbf{s}_1 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{11}$$

Further iterations reveal that $L^{2k} = L^2$ and $L^{2k+1} = L^3$ for $k = 1, 2, \ldots$. Hence, the algorithms constructed in [15] can find all steady states and cycles by inspecting L^2 and L^3. However, one of the main drawbacks of this approach is the requirement for *always* checking all 2^n states to construct L. Hence, the algorithm is limited to solving only small or moderate-size Boolean networks only.

On the other hand, the approach we propose does not require explicitly checking 2^n states or constructing the matrix L. As shown in Figure 2, the recursive self-composition provides the converged cyclic logic without checking the 2^3 states. The following methods and results in the paper do not have links to the semi-tensor approach. We use semi-tensor notations to improve the clarity of proofs or explanations.

B. Main Results

Assumption 1 (Synchronous Update): All states in the Boolean network given by (2) are updated synchronously. The states in the right-hand-side of (2) is the one at the same step.

Definition 1 (Recursive Self-Composite): The p-times recursive self-composite of the Boolean network is given by

$$\begin{aligned} \mathbf{x}(p) &= \mathbf{f}[\mathbf{x}(p-1)] = \mathbf{f}[\mathbf{f}(\mathbf{x}(p-2))] = \ldots \\ &= \mathbf{f}[\mathbf{f}(\mathbf{f} \ldots (\mathbf{f}(\mathbf{x}(0))))] \\ &= \underbrace{\mathbf{f} \circ \mathbf{f} \circ \mathbf{f} \circ \ldots \circ \mathbf{f}}_{p-\text{times}}[\mathbf{x}(0)] = \mathbf{f}^p[\mathbf{x}(0)] \end{aligned} \tag{12}$$

where p is a positive integer.

Theorem 1 (Convergence of Recursive Self-Composition): All recursive self-composition of the synchronous Boolean network given by (2) converges to a steady-state logic or a cyclic logic, i.e,

$$\mathbf{f}^{p^*+r}(\mathbf{x}) = \mathbf{f}^{p^*+r+\ell^*}(\mathbf{x}) \tag{13}$$

where p^* between 0 and 2^n is the minimum number of recursions when the logic starts repeating itself, r is a non-negative integer and ℓ^* between 0 and 2^n is the period of the logic cycle, which is equal to the least common multiple of all state cycle lengths.

Proof: Deterministic synchronous update Boolean networks have a finite number of states, i.e., 2^n, any initial state repeats the same state at longest in 2^n steps. In the case of converging to a steady state, there exists the smallest integer $p_s \in [0, 2^n]$ such that $\mathbf{x}(p_s) = \mathbf{x}(p_s + r)$ for all integer $r \geq 0$. In the case of converging to a state cycle with the period ℓ, there exists the smallest integer $p_c \in [0, 2^n]$ such that $\mathbf{x}(p_c) = \mathbf{x}(p_c + \ell)$.

Let p^* be the largest value among p_s and p_c for all steady states and cycles and ℓ^* be the period of a state cycle. Rewrite the equation for the cycle as $\mathbf{x}(p^*) = \mathbf{x}(p^* + \ell^*)$. It leads to

$$\mathbf{f}^{p^*}[\mathbf{x}(0)] = \mathbf{f}^{p^*+\ell^*}[\mathbf{x}(0)] \tag{14}$$

From the number of compositions larger than p^*, the state belongs to one of the states in the longest state cycle, the following equation is satisfied for $r = 1, 2, \ldots$,

$$\mathbf{x}(p^* + r) = \mathbf{x}(p^* + \ell^* + r) \tag{15}$$

and this provides

$$\mathbf{f}^{p^*+r}[\mathbf{x}(0)] = \mathbf{f}^{p^*+\ell^*+r}[\mathbf{x}(0)] \tag{16}$$

for all $\mathbf{x}(0)$ in the period state cycle. Repeating the same procedure for each state cycle provides the common ℓ^* to be the least common multiple of all ℓ^*.

Finally, assume that the logic cycle is strictly longer than ℓ^*, say $\ell^* + 1$, and the initial state is given by $\mathbf{f}^{p^*}[\mathbf{x}(0)]$. At the $(\ell^* + 1)$-th step, the logic returns to the first logic, $\mathbf{f}^{p^*}[\mathbf{x}(0)]$, and provides the same state as the initial state. This contradicts the definition of a state cycle by making the two states identical. Therefore, the longest logic cycle must be equal to ℓ^*. ■

When ℓ^* is equal to 0, it corresponds to the case that there is only one steady-state logic. If ℓ^* is equal to 3, three update rules switch between them. For instance, the Boolean network given in (4) has p^* equal to 2, when the logic repetition begins to start, and ℓ^* equal to 2, which is the period of logic cycles.

Remark 1: In the worst case, if p^* is equal to 2^n, the computational cost to find a repeating logic is at least as expensive as the exhaustive search. This is inevitable in exact algorithms for solving NP-hard problems.

Theorem 2 (Longest Cycle Upper Bound): All cycle lengths of every Boolean logic cannot be longer than the length of the converged logic cycle, ℓ^*.

Proof: Let us assume there exists a cycle of the period, m, strictly longer than ℓ^*. Let $\mathbf{s}_{p^*}$ be the state for the first time arrived in the cycle at p^*-step from the state $\mathbf{s}_{p^*-1}$, which is not in the cycle, and the cycle propagates as follows:

$$\begin{array}{ccccccc}
\mathbf{s}_{p^*-1} & \xrightarrow{L} & \mathbf{s}_{p^*} & \xrightarrow{L} & \mathbf{s}_{p^*+1} & \xrightarrow{L} & \mathbf{s}_{p^*+2} \\
 & & & & & & \downarrow L \\
\mathbf{s}_{p^*+\ell^*} & \xleftarrow{L} & \mathbf{s}_{p^*+\ell^*-1} & \xleftarrow{L} & \cdots & \xleftarrow{L} & \mathbf{s}_{p^*+3} \\
L \downarrow & & & & & & \\
\mathbf{s}_{p^*+\ell^*+1} & \xrightarrow{L} & \cdots & \xrightarrow{L} & \mathbf{s}_{p^*+m-1} & \xrightarrow{L} & \mathbf{s}_{p^*}
\end{array}$$

where $\mathbf{s}_{p^*}$ in the last line is equal to the one in the first line and the cycle repeats. By the definition of cycle, $\mathbf{s}_i \neq \mathbf{s}_j$ for $i \neq j$. Also, notice that as $\mathbf{s}_{p^*+m} = \mathbf{s}_{p^*}$ and $\mathbf{f}^{p^*+m}(\mathbf{x}_0) = \mathbf{f}^{p^*}(\mathbf{x}_0)$ m must be an integer multiple of ℓ^*.

As m is strictly greater than ℓ^*, consider $(p^* + \ell^* + 1)$-step using the composition logic as follows:

$$\mathbf{s}_{p^*+\ell^*+1} = L\mathbf{s}_{p^*+\ell^*} \rightarrow \mathbf{x}(p^* + \ell^* + 1) = \mathbf{f}^{p^*+\ell^*+1}[\mathbf{x}(0)]$$

where $\mathbf{x}(p^* + \ell^* + 1)$ corresponds to $\mathbf{s}_{p^*+\ell^*+1}$. By Theorem 1, the following equality satisfies

$$\mathbf{f}^{p^*+\ell^*+1}[\mathbf{x}(0)] = \mathbf{f}^{p^*+1}[\mathbf{x}(0)] \tag{17}$$

Hence,

$$\mathbf{s}_{p^*+\ell^*+1} = \mathbf{s}_{p^*+1} \tag{18}$$

This contradicts $\mathbf{s}_i \neq \mathbf{s}_j$ for $i \neq j$ in the cycle. Therefore, no cycle can have a longer period than ℓ^*.

As the interval $p^* + k \leq i$ or $j \leq p^* + m - k$ with $i \neq j$ and a positive integer k shifts the starting point of the cycle k-step forward, the proof for the shifted interval is the same as the one for $k = 0$ shown above. ■

The two logics, $\ell^* = 2$, switch between them in the example shown in Figure 2, where p^* is equal to 2. If there is a cycle with a period of 4, the states in the cycle ($\mathbf{s}_0$, $\mathbf{s}_1$, $\mathbf{s}_2$ and $\mathbf{s}_3$) must be different from each other. Let each state in the cycle, $\mathbf{s}_r$, correspond to $\mathbf{f}^{p^*+r}[\mathbf{x}(0)]$ for r equal to 0, 1, 2, or 3. The left-hand side of (17) becomes $\mathbf{f}^{2+2+1}[\mathbf{x}(0)] = \mathbf{f}^{2+3}[\mathbf{x}(0)]$, whose corresponding state is $\mathbf{s}_3$. The right-hand side of (17), becomes $\mathbf{f}^{2+1}[\mathbf{x}(0)]$ corresponding to $\mathbf{s}_1$. Hence, $\mathbf{s}_3$ is equal to $\mathbf{s}_1$, and this cannot be allowed in the cycle. Similarly, we can show that $\mathbf{s}_0$ is equal to $\mathbf{s}_1$. This implies that the period of a probable cycle is 2.

Definition 2 (Kernel States Set): The kernel states set, $\mathbb{K}$, of the synchronous Boolean network, (2), includes all steady-states and the states belonging to cycles.

For instance, all the states indicated in red in Figure 1 are the kernel states of the network and $\mathbb{K} = \{(000), (001), (010), (011), (111)\}$ or equivalently $\mathbb{K} = \{s_0, s_1, s_2, s_3, s_7\}$.

Theorem 3 (Kernel States Set of Converged Logic): The one-step propagated state, $\mathbf{x}(1)$, by the converged logic in (13), i.e., $\mathbf{x}(1) = \mathbf{f}^{p^*+r}[\mathbf{x}(0)]$, converges the same kernel states set of the original Boolean network for any fixed non-negative integer r.

Proof: If a steady-state is absent in the kernel set of a converged logic, then it contradicts the property of steady-states. Hence, the proof for steady-states cases becomes trivial. Let us consider a cycle of the period, ℓ^*, as follows:

$$\mathbf{s}_0 \underbrace{\rightarrow}_{L\mathbf{s}_0} \mathbf{s}_1 \underbrace{\rightarrow}_{L\mathbf{s}_1} \mathbf{s}_2 \underbrace{\rightarrow}_{L\mathbf{s}_2} \cdots \underbrace{\rightarrow}_{L\mathbf{s}_{\ell^*-1}} \mathbf{s}_{\ell^*} \tag{19}$$

Let the initial state, $\mathbf{x}(0)$, correspond to $\mathbf{s}_0$. And, it propagates p^* steps as follows:

$$\begin{aligned}\mathbf{x}(1) = \mathbf{f}[\mathbf{x}(0)] \rightarrow \mathbf{x}(2) = \mathbf{f}[\mathbf{x}(1)] = \mathbf{f}^2[\mathbf{x}(0)] \rightarrow \ldots \\ \ldots \rightarrow \mathbf{x}(p^*) = \mathbf{f}[\mathbf{x}(p^* - 1)] = \mathbf{f}^{p^*}[\mathbf{x}(0)]\end{aligned} \tag{20}$$

As $\mathbf{x}(0)$ starts in the cycle, all propagated states are in the cycle. Hence, $\mathbf{x}(p^*)$ is equal to one of the states in the cycle. Specifically, $\mathbf{x}(p^*)$ is equal to the state corresponding to $\mathbf{s}_{r^*}$, where $r^* = p^* - \ell^* q$, which is in $[0, \ell^*]$, q is the maximum integer such that $\ell^* q$ is less than or equal to p^*.

Without loss of the generality, let us assume that r^* is equal to 2, i.e., $\mathbf{x}(p^*)$ corresponds to $\mathbf{s}_2$. It implies that: $\mathbf{s}_2$ is an element of the kernel state set of $\mathbf{f}^{p^*}$, $\mathbf{s}_3$ is an element of the kernel state set of $\mathbf{f}^{p^*+1}$ and so forth.

Let us choose the initial state, $\mathbf{x}(0)$ corresponding to $\mathbf{s}_1$ and repeat the same procedure. Then, r^* becomes 3 and this results in: $\mathbf{s}_3$ is an element of the kernel state set of $\mathbf{f}^{p^*}$, $\mathbf{s}_4$ is an element of the kernel state set of $\mathbf{f}^{p^*+1}$ and so forth.

For the shorter cycles less than the period ℓ^*, the same steps provide the proof that all the cycle states must be in the kernel state of each of the converged logic. Therefore, the converged logic includes all states in the cycles. ■

Theorem 4 (Longest Length Cycle): If there exists only one state cycle, its period is equal to the period of the logic cycle, ℓ^*.

Proof: By Theorem 3, the range set of every converged logic is identical with each other as the kernel set, $\mathbb{K}$. Hence, once the logic converges to the logic cycle, whose period is ℓ^*, the mapping from $\mathbb{K}$ to $\mathbb{K}$ repeats ℓ^* times. Each of the mappings must be different from each other. Otherwise, the existence of the logic cycle equal to ℓ^* is violated. In addition, due to the periodicity of the logic cycles, the ℓ^*-th mapping brings the states back to the states mapped by the first logic cycle. ■

Definition 3 (Kernel Logic): The Kernel logic, $\mathbf{f}^{p^*+k^*}$ is the converged logic having the same steady states and cycles as the original Boolean logic, where k^* is an integer between 0 and $\ell^* - 1$.

For instance, the converged logic in the right-hand side of Figure 2 has the same three steady states and one cycle as the original network given by (4).

Theorem 5 (Existence of Kernel Logic): For every Boolean network, there exists at least one kernel logic among the converged logic cycles.

Proof: Given that there are ℓ^* number of converged logic, $\mathbf{f}^{p^*+k}$, where k is an integer from 0 to $\ell^* - 1$, take t-time self-composites for a fixed k logic as follows:

$$\underbrace{\mathbf{f}^{p^*+k} \circ \ldots \circ \mathbf{f}^{p^*+k} \circ \mathbf{f}^{p^*+k}}_{t-\text{times}}[\mathbf{x}(0)] = \mathbf{f}^{t(p^*+k)}[\mathbf{x}(0)] \quad (21)$$

$p^* + k$ can be expressed as

$$p^* + k = m\ell^* + r \quad (22)$$

where m is the largest integer satisfying $p^* + k \geq m\ell^*$, where r can be any integer between 0 and $\ell^* - 1$ as k is between 0 and $\ell^* - 1$. Among the logic corresponding to each r, we choose the logic for $r = 1$ and multiply t as follows:

$$t(p^* + k) = tm\ell^* + t \quad (23)$$

Hence, we can cover all integers from 0 to $\ell^* - 1$ by varing t. Therefore, it covers all ℓ^* cyclic logic and there exists always at least one kernel logic. ■

Remark 2 (Leaping & Filling): One of the ways to speed up the self-composition iteration and possibly increase the chance to avoid large string-length explosions is leaping by performing larger-step composition instead of a one-step composition. First,

$$\mathbf{x}(k+2) = \mathbf{f}^2[\mathbf{x}(k)] = \mathbf{g}[\mathbf{x}(k)]$$

is obtained. Secondly,

$$\mathbf{x}(k+4) = \mathbf{g}[\mathbf{x}(k+2)] = \mathbf{g}^2[\mathbf{x}(k)] = \mathbf{h}[\mathbf{x}(k)]$$

then,

$$\mathbf{x}(k+8) = \mathbf{h}[\mathbf{x}(k+4)] = \mathbf{h}^2[\mathbf{x}(k)]$$

and we continue until the logic converges. Once the logic converges, we apply $\mathbf{f}(\cdot)$ repeatedly and obtain the logic between the leaps. For instance, $\mathbf{h}[\mathbf{x}(k)]$ converges and $\mathbf{h}^2[\mathbf{x}(k)]$ is equal to $\mathbf{h}[\mathbf{x}(k)]$. Then, the logic for $\mathbf{x}(k+5)$, $\mathbf{x}(k+6)$ and $\mathbf{x}(k+7)$ are obtained by the filling sequence as follows:

$$\mathbf{x}(k+5) = \mathbf{f}[\mathbf{x}(k+4)] = \mathbf{f}\{\mathbf{h}[\mathbf{x}(k)]\}$$
$$\mathbf{x}(k+6) = \mathbf{f}[\mathbf{x}(k+5)] = \mathbf{f}^2\{\mathbf{h}[\mathbf{x}(k)]\}$$
$$\mathbf{x}(k+7) = \mathbf{f}[\mathbf{x}(k+6)] = \mathbf{f}^3\{\mathbf{h}[\mathbf{x}(k)]\}$$

Algorithms 1 and 2 provide the summaries of the procedures, and Algorithm 3 shows the one-step composition process switches to the leaping and filling process when a large string-length explosion occurs. The way of choosing some values in

Algorithm 1 Leaping(n_{leap} : the number of leaping)

Set $p = 1$, $q = 1$, $n_{\text{loop}} = 0$
while $n_{\text{loop}} < n_{\text{leap}}$ **do**
  Set $p \leftarrow 2p$
  Substituting $\mathbf{x}(k+q) = \mathbf{f}^q[\mathbf{x}(k)]$ into:
    $\mathbf{x}(k+p) = \mathbf{f}^p[\mathbf{x}(k+q)]$
  Set $q \leftarrow p$ and $n_{\text{loop}} \leftarrow n_{\text{loop}} + 1$
end while
return $\mathbf{x}(k+p) = \mathbf{f}^p[\mathbf{x}(k)]$, where $p = 2^{n_{\text{leap}}}$

Algorithm 2 Filling(n_{leap})

Set $p = n_{\text{leap}}/2 + 1$
while $p < n_{\text{leap}}$ **do**
  Obtain $\mathbf{x}(k+p) = \mathbf{f}^p[\mathbf{x}(k)]$
  $p \leftarrow p + 1$
end while
return $\mathbf{x}(k+p) = \mathbf{f}^p[\mathbf{x}(k)]$
  for all $p \in \{1, 2, \ldots, n_{\text{leap}}/2 - 1\} + n_{\text{leap}}/2$

Algorithm 3 Switching($t_{\max}$: maximum computing time)

Set n_w, $i \leftarrow 0$ and $p \leftarrow 1$
while True **do**
  Set n_f and $j \leftarrow 2$
  ——(p-step recursion)——
  for $j < n_f$ **do**
    $n \leftarrow j \times p$
    $\mathbf{x}(n) = \mathbf{f}^n[\mathbf{x}(0)]$
    $j \leftarrow j + 1$
    if (Computing time for p-step recursion) $> t_{\max}$ or logic converges **then**
      break the for-loop
    end if
  end for
  ——(Leaping)————
  if logic converges **then**
    break the while-loop
  end if
  Set n_{leap} and Leaping(n_{leap})
  if (Computing time for leaping) $> t_{\max}$ **then**
    $p \leftarrow 2^{n_{\text{leap}}}$
  end if
  $i \leftarrow i + 1$
  if $i > n_w$ **then**
    Declare the logic failed to converge
    break the while-loop
  end if
end while
——(Filling)————
if logic converges **then**
  Filling(n_{leap})
  if (Computing time for filling) $> t_{\max}$ **then**
    Declared the logic failed to converge
  end if
end if

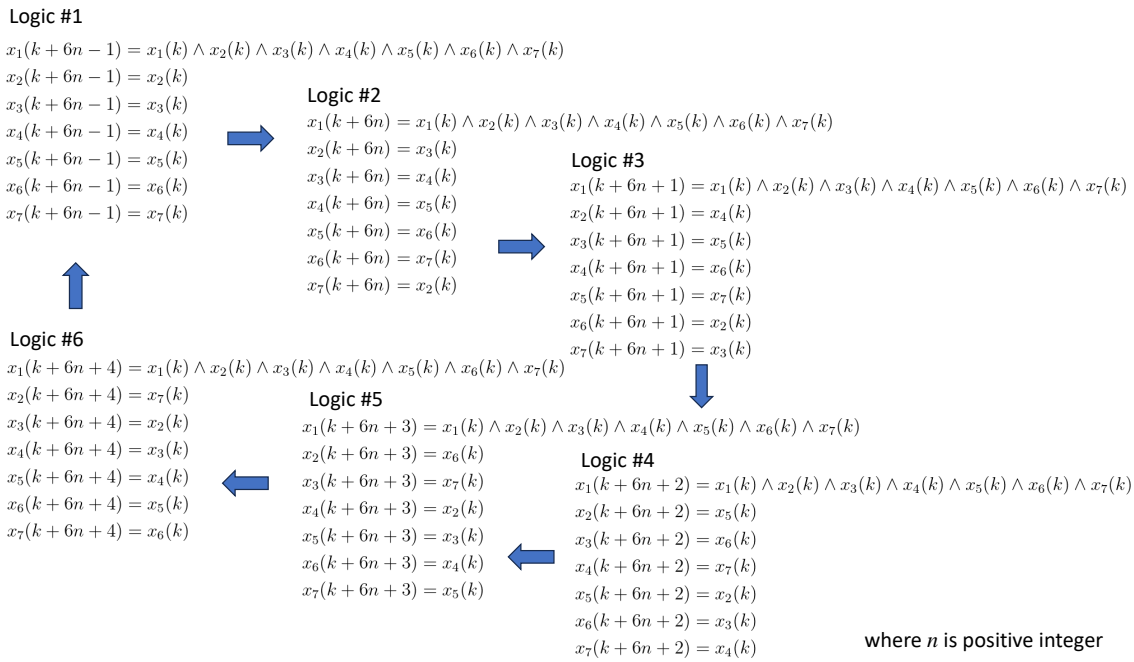

Fig. 3. The Boolean logic in (24) cycles between six update rules.

the switching algorithm is ad-hoc or heuristic, for example, the number of p-step iteration, n_f, and the number of leaping, n_{leap}. There are places to automatize the optimal selection of these values.

III. Examples & Discussion

Example 1 (Longer feedback network): The following Boolean network is an extended network of (4) having a longer feedback chain between x_2 and x_7:

$$x_1(k+1) = x_1(k) \wedge x_2(k) \tag{24a}$$
$$x_2(k+1) = x_3(k) \tag{24b}$$
$$x_3(k+1) = x_4(k) \tag{24c}$$
$$x_4(k+1) = x_5(k) \tag{24d}$$
$$x_5(k+1) = x_6(k) \tag{24e}$$
$$x_6(k+1) = x_7(k) \tag{24f}$$
$$x_7(k+1) = x_2(k) \tag{24g}$$

By the recursive compositions, the logic converges at $p^* = 5$ to the cyclic logic whose period, ℓ^*, is equal to 6 as shown in Figure 3. It has three steady-states, a cycle with period 2, two cycles with period 3 and nine cycles with period 6. The number of elements in the kernel states set is 65 ($= 3+1\times2+2\times3+9\times6$). Among the six logics in the cycle, Logic #2 and #6 in Figure 3 are the kernel logic. By inspecting the right-hand side of the converged logic, the common characteristics found are as follows: x_1 equal to 1 is a rare event, and the other states equal to 0 or 1 have the same chance. This coincides with the fact that there is only one kernel state with x_1 equal to 1 among the 65 kernel states. Hence, we would be able to design numerical and lab experiments to find rare events, which would be challenging to find even in numerical experiments based on Monte Carlo type random simulations.

Example 2 (T-cell receptor network): A T-cell receptor Boolean network model in [20] having 37 states with 3 control inputs is given by

$$\begin{aligned}
\text{AP1}(k+1) &= \text{Fos}(k) \wedge \text{Jun}(k),\ \text{Ca}(k+1) = \text{IP3}(k)\\
\text{Calcin}(k+1) &= \text{Ca}(k),\ \text{cCbl}(k+1) = \text{ZAP70}(k)\\
\text{CRE}(k+1) &= \text{CREB}(k),\ \text{CREB}(k+1) = \text{Rsk}(k)\\
\text{DAG}(k+1) &= \text{PLCg}^*(k),\ \text{ERK}(k+1) = \text{MEK}(k)\\
\text{Fos}(k+1) &= \text{ERK}(k)\\
\text{Fyn}(k+1) &= [\text{Lck}(k) \wedge \text{CD45}(k)]\\
&\quad \vee [\text{TCR}^+(k) \wedge \text{CD45}(k)]\\
\text{Gads}(k+1) &= \text{LAT}(k),\ \text{Grb2Sos}(k+1) = \text{LAT}(k)\\
\text{IKKbeta}(k+1) &= \text{PKCth}(k),\ \text{IP3}(k+1) = \text{PLCg(act)}(k)\\
\text{Itk}(k+1) &= \text{SLP76}(k) \wedge \text{ZAP70}(k)\\
\text{IkB}(k+1) &= \neg\text{IKKbeta}(k),\ \text{JNK}(k+1) = \text{SEK}(k)\\
\text{Jun}(k+1) &= \text{JNK}(k),\ \text{LAT}(k+1) = \text{ZAP70}(k)\\
\text{Lck}(k+1) &= \neg\text{PAGCsk}(k) \wedge \text{CD45}(k) \wedge \text{CD4}(k)\\
\text{MEK}(k+1) &= \text{Raf}(k),\ \text{NFAT}(k+1) = \text{Calcin}(k)\\
\text{NFkB}(k+1) &= \neg\text{IkB}(k),\ \text{PKCth}(k+1) = \text{DAG}(k)\\
\text{PLCg}^*(k+1) &= [\text{Itk}(k) \wedge \text{PLCg}^+(k) \wedge \text{SLP76}(k)\\
&\quad \wedge \text{ZAP70}(k)] \vee [\text{PLCg}^+(k) \wedge \text{Rlk}(k)\\
&\quad \wedge \text{SLP76}(k) \wedge \text{ZAP70}(k)]\\
\text{PAGCsk}(k+1) &= \text{Fyn}(k) \vee \neg\text{TCR}^+(k)\\
\text{PLCg}^+(k+1) &= \text{LAT}(k),\ \text{Raf}(k+1) = \text{Ras}(k)\\
\text{Ras}(k+1) &= \text{Grb2Sos}(k) \vee \text{RasGRP1}(k)\\
\text{RasGRP1}(k+1) &= \text{DAG}(k) \wedge \text{PKCth}(k)\\
\text{Rlk}(k+1) &= \text{Lck}(k),\ \text{Rsk}(k+1) = \text{ERK}(k)
\end{aligned}$$

$$\begin{aligned}
\text{SEK}(k+1) &= \text{PKCth}(k),\ \text{SLP76}(k+1) = \text{Gads}(k)\\
\text{TCR}^{+}(k+1) &= \neg\text{cCbl}(k) \wedge \text{TCRlig}(k)\\
\text{TCR}^{\dagger}(k+1) &= \text{Fyn}(k) \vee [\text{Lck}(k) \wedge \text{TCR}^{+}(k)]\\
\text{ZAP70}(k+1) &= \neg\text{cCbl}(k) \wedge \text{Lck}(k) \wedge \text{TCR}^{\dagger}(k)
\end{aligned}$$

where $(\cdot)^{+}$ represents the binding status, $(\cdot)^{*}$ denotes being activated, $(\cdot)^{\dagger}$ indicates phosphates, CD45, CD4 and TCRlig are the three control inputs and there are four biomolecular species to be considered as the outputs of the networks, which are AP1, CRE, NFAT and NFkB. In [18][1], this T-cell Boolean network was used for structural controllability analysis to design a feedback controller.

*Case 1 (*CD45=1, CD4=1, TCRlig=1*):* AP1 and NFAT converges to 0, NFkB converges to 1 and CRE switches between the following six update rules:

$$\begin{aligned}
\text{CRE}(k+6n{+}7) &= \text{cCbl}(k) \wedge \text{PAGCsk}(k) \wedge \neg\text{ZAP70}(k)\\
&\quad \wedge [\text{Fyn}(k) \vee \neg\text{TCR}^{+}(k)] \qquad (25a)\\
\text{CRE}(k+6n{+}8) &= \text{ZAP70}(k) \wedge [\text{Fyn}(k) \vee \neg\text{TCR}^{+}]\\
&\quad \wedge [\text{cCbl}(k) \vee \text{Lck}(k) \vee \text{TCR}^{+}(k)]\\
&\quad \wedge [\text{cCbl}(k) \vee \neg\text{Lck}(k) \vee \neg\text{TCR}^{\dagger}(k)] \qquad (25b)\\
\text{CRE}(k+6n{+}9) &= \text{Lck}(k) \wedge \text{TCR}^{\dagger}(k) \wedge \neg\text{cCbl}(k)\\
&\quad \wedge [\text{PAGCsk}(k) \vee \text{ZAP70}(k) \vee \neg\text{Fyn}(k)]\\
&\quad \wedge [\text{PAGCsk}(k) \vee \text{ZAP70}(k) \vee \neg\text{TCR}^{+}(k)] \qquad (25c)\\
\text{CRE}(k+6n{+}10) &= \neg\text{PAGCsk}(k) \wedge \neg\text{ZAP70}(k)\\
&\quad \wedge [\text{Fyn}(k) \vee \text{Lck}(k)] \wedge [\text{Fyn}(k) \vee \text{TCR}^{+}(k)]\\
&\quad \wedge [\text{Fyn}(k) \vee \text{TCR}^{\dagger}(k)] \wedge [\text{Fyn}(k) \vee \neg\text{cCbl}(k)] \qquad (25d)\\
\text{CRE}(k+6n{+}11) &= \text{TCR}^{+}(k) \wedge \neg\text{Fyn}(k)\\
&\quad \wedge [\text{cCbl}(k) \vee \neg\text{Lck}(k) \vee \neg\text{TCR}^{\dagger}(k)] \qquad (25e)\\
\text{CRE}(k+6n{+}12) &= \neg\text{cCbl}(k) \wedge \neg\text{Lck}(k) \wedge \neg\text{TCR}^{+}(k)\\
&\quad \wedge [\text{PAGCsk}(k) \vee \text{ZAP70}(k) \vee \neg\text{Fyn}(k)] \qquad (25f)
\end{aligned}$$

where n is positive integer. Based on this finding, an additional feedback control input would be designed to derive CRE towards desired states. A total of 21 states update logic out of the 37 states converge to a period of 6 switching logic. The remaining 16 states converge to steady states of either 0 or 1. The state space shrinks from 2^{37}(137 billion) to 2^{21}(2 million), which is only 0.0015% of the original size of the state space.

Case 2 (at least one of CD45, CD4 *or* TCRlig *equal to 0):* AP1, CRE and NFAT converge to 0 and NFkB converges to 1. There is no possibility of introducing further control structures to change the outputs.

These analyses lead the problem space from originally computationally infeasible to feasible ranges. In addition, they clearly show what states can or cannot be controlled and what the update-rule structures of states to be controlled are.

From a biological perspective, when foreign antigens are presented to T cell receptors (TCR) and co-receptors (CD4), along with the involvement of receptor-type protein tyrosine phosphatase (CD45) [20], this triggers an immune response signaling cascade that includes the Ras-Raf-Mek-Erk pathway. This, in turn, leads to the transcriptional activation of numerous immune-related genes, such as IL-2, IL-6, IL-10, TNF-α, and others [21]. The promoters of these genes commonly contain a DNA target sequence known as the cAMP-responsive element (CRE), to which transcription factors belonging to the CREB family (CREB) can specifically bind and initiate transcription.

To counterbalance the risk of an overactive immune response that could potentially result in autoimmunity, a negative feedback mechanism is in place. This mechanism is primarily mediated by the E3 ubiquitin ligase (cCbl) [22] and Csk-associated adaptor PAG (PAGCsk) [23]. These two proteins ensure immune response homeostasis, the former by facilitating the degradation of key signaling proteins including the activated protein tyrosine kinase (ZAP70) [22], and the latter by inactivating Src family kinases (Lck, Fyn) [23]. Within the T-cell receptor network, with all control inputs set to 1, two negative feedback loops (cCbl-ZAP70 and PAGCsk-Lck-Fyn) continuously operates, resulting in oscillations between 0 and 1, with the period 2 and 3, respectively. These two negative feedback loops are interconnected with two nodes (TCR^{+} and $\text{TCR}^{\dagger}$), resulting in the complex dynamics of the coupled feedback loops expected to have a period of 6.

This regulatory process corresponds to the results of *Case 1 (*CD45=1, CD4=1, TCRlig=1*)*, where the recursive self-composite logic of CRE switches between the six update rules, (25). In (25), Ras-Raf-Mek-Erk pathway between CRE and upstream feedback loops is omitted, while only the coupled feedback regulations are represented as a switching for the presence or absence of the negation operator in front of cCbl(k), ZAP70(k), PAGCsk(k), Lck(k) and Fyn(k). In the results of *Case 2*, on the other hand, when any one of the input controls equals to 0, multiple feedback loops will no longer be able to operate, leading to ZAP70, Lck, and Fyn keeping inactivated so that CRE converges to 0. This signifies that the immune response signalling cascade is deactivated, rendering the homeostatic effect of negative feedback unnecessary.

The cyclic logics of CRE, (25), obtained by the proposed recursive algorithm extract an intuitive representation of the long-term oscillatory dynamics of the output of the T-cell Boolean network model.

Example 3 (Cancer signaling network): In the study by Fumiã et al. [10], a Boolean network model was developed to represent human tumorigenesis. This model encompasses 90 states of proteins within cancer signaling pathways, alongside 6 control inputs – Mutagen, GFs (growth factors), Nutrients, TNFα, Hypoxia and Gli. The Boolean functions in the network were initially formulated using algebraic operators ($+$, $-$) and $\text{sgn}(\cdot)$ – a thresholding function in which $\text{sgn}(x) = 0$ for $x \leq 0$ and 1 for $x > 0$. As this algebraic representation is less straightforward for recursive self-composition, we opted to transform the Boolean functions into an equivalent form utilizing the Boolean operations, while preserving their original truth tables, the Boolean model is available to download as indicated in Supplementary Material.

The network outputs include two bio-molecules, Glut1 and Lactic acid, as well as two virtual nodes that represent cellular

[1]The T-cell model in [18] adopted from [20] includes a few typos.

phenotypes – Apoptosis and DNA repair. In addition, the other cellular phenotype, Proliferation, was defined by examining the long-term dynamics of the network, particularly the activation sequence of cyclin nodes. In [24], attractor-transition analysis of this network was carried out to investigate the critical transition of tumorigenesis along with the accumulation of driver mutations.

The update logic for Cyclins D has the longest string. Its update rule has 23k characters. And, the first three self-compositions make the length increase exponentially, i.e., 149k, 34M and 293M. Applying the self-composition procedure to the original network produces long string chains of the updated rules. The lengths of the strings are too long to be handled by most digital computers. As analyzing Boolean networks is known to be an NP-hard problem [17], finding some networks producing large strings beyond the current computer calculation speed and memory capacity is not surprising.

To restrict the network to the condition with normoxic (normal oxygen level), sufficient growth factors and nutrient-rich five input states are set as follows: Mutagen = 0, GFs = 1, Nutrients = 1, TNFα = 0, Hypoxia = 0 and Gli = 0. In [10], it is shown that the cell enters into the proliferation cycle with a period of 7.

p53, Cyclins A and Cyclins D are identified by trial and error as the additional control inputs to change the phenotype. Pinning the three states to 1 and applying the self-compositions make the logic length become too large again before it converges. By applying the leaping every 4 iterations, i.e., starting from $\mathbf{x}(k+1)$, we obtain $\mathbf{x}(k+2)$, $\mathbf{x}(k+3)$, $\mathbf{x}(k+4)$ and $\mathbf{x}(k+5)$. Then, leaping from $\mathbf{x}(k+5)$, we obtain $\mathbf{x}(k+10)$, $\mathbf{x}(k+15)$ and $\mathbf{x}(k+20)$, and it converges at $\mathbf{x}(k+15)$. And, applying the filling procedure from $\mathbf{x}(k+16)$ to $\mathbf{x}(k+19)$ we confirm the logic converges to a single logic.

The maximum string length of the updated rules for each iteration is as follows: 17k, 29k, 60k, 65k, 64k, 10 and 10, where the corresponding longest string states are TSC1/2, Cytoc/APAF1, Cytoc/APAF1, GSH, eEF2k, E2F and E2F, respectively. It converges to a steady state instead of the cycle and the phenotype changes from proliferation to apoptosis. More interestingly, all states converge to 0 or 1 regardless of the initial conditions except E2F$(k+15)$, a transcription factor for cell-cycle regulation genes, which converges to

$$\text{E2F}(k+15) = \text{E2F}(k) \wedge [\neg \text{Rb}(k)]$$

where Rb is retinoblastoma protein.

The example demonstrates the power of the proposed method in finding a hidden simple logic behind the complex networks. Such finding can help to identify further important drug target candidates to be developed for efficient cancer therapeutic strategies.

Discussion: We have demonstrated the usage of the proposed method based on the recursive self-composition of Boolean networks with the three examples above. There might be some hidden fundamental links between the structural properties of Boolean networks and p^* or the kernel logic. The proposed method would fail to converge sometimes to a simple logic because the limitations present in the computers or the converged logic to be found is not simple. What structural characteristics of the Boolean networks give rise to either a simple or complex converged logic is a hard but important question to answer.

How these mathematical structures of the Boolean networks map to Biological networks is another fundamental question. Biological interactions are inherently spatiotemporal stochastic [25]. Each molecular species seems to interact with many others. Meanwhile, measurement technologies make steady progress and high-throughput data available [26]. Boolean network is an ideal model for large-scale biological networks with high-throughput data [27]. The method proposed increases the capability of analysing larger-size Boolean networks. Although we introduced our method based on deterministic Boolean network models with synchronous updates, the proposed approach can be extended to other forms of Boolean network models with asynchronous or rule-based update schemes

The Boolean network model has a limitation in describing the continuous biological variables by simplified discrete values of either on or off. Despite such a limitation, it has been well-proven that Boolean network models are still useful as they can capture essential dynamics of various biological systems by representing biologically important phenotypes in terms of attractor states [12], [28], [29] through many experimental demonstrations [13], [14], [30].

IV. Conclusions & Future Works

We present the recursive self-composite approach to reveal the hidden characteristics of Boolean networks. Most importantly, we found the cyclic nature of synchronous Boolean update rules. This is the first time that the converging nature of the Boolean logic dynamics is unveiled explicitly. We also found several interesting properties of the converged logic: the existence of kernel logic and its relationship with the length of periodic cycles in the state space. There might be many other hidden fundamental structures of the Boolean network.

Finding the category of Boolean networks that can be characterized by the properties found by the recursive procedure, and the essential relationship between the repeating logic and biological phenomena to control the behaviour of Boolean networks and extending the approaches to asynchronous Boolean networks are of immediate interest for future study.

Supplementary Material

The Boolean network model of the cancer signalling network in Python is available to download at the following link: https://github.com/myjr52/Fumia_cancer_network_boolean_model

Acknowledgment

The authors would like to thank the support of the Cheney fellowship. The research was initiated in June 2023 by the visit of KHC to the University of Leeds as one of the Cheney fellowship activities, hosted by JK.

REFERENCES

[1] S. Kauffman, "Homeostasis and differentiation in random genetic control networks," *Nature*, vol. 224, no. 5215, pp. 177–178, 1969.

[2] S. Huang, G. Eichler, Y. Bar-Yam, and D. E. Ingber, "Cell fates as high-dimensional attractor states of a complex gene regulatory network," *Physical Review Letters*, vol. 94, no. 12, p. 128701, 2005.

[3] P. Zhou, S. Wang, T. Li, and Q. Nie, "Dissecting transition cells from single-cell transcriptome data through multiscale stochastic dynamics," *Nature communications*, vol. 12, no. 1, p. 5609, 2021.

[4] X. Kang and C. Li, "A dimension reduction approach for energy landscape: identifying intermediate states in metabolism-emt network," *Advanced Science*, vol. 8, no. 10, p. 2003133, 2021.

[5] O. S. Rukhlenko, M. Halasz, N. Rauch, V. Zhernovkov, T. Prince, K. Wynne, S. Maher, E. Kashdan, K. MacLeod, N. O. Carragher *et al.*, "Control of cell state transitions," *Nature*, vol. 609, no. 7929, pp. 975–985, 2022.

[6] F. Chen, Y. Bai, and C. Li, "Estimation of non-equilibrium transition rate from gene expression data," *Briefings in Bioinformatics*, vol. 24, no. 3, p. bbad113, 2023.

[7] Ö. Sahin, H. Fröhlich, C. Löbke, U. Korf, S. Burmester, M. Majety, J. Mattern, I. Schupp, C. Chaouiya, D. Thieffry *et al.*, "Modeling erbb receptor-regulated g1/s transition to find novel targets for de novo trastuzumab resistance," *BMC Systems Biology*, vol. 3, no. 1, pp. 1–20, 2009.

[8] S. Collombet, C. Van Oevelen, J. L. Sardina Ortega, W. Abou-Jaoudé, B. Di Stefano, M. Thomas-Chollier, T. Graf, and D. Thieffry, "Logical modeling of lymphoid and myeloid cell specification and transdifferentiation," *Proceedings of the National Academy of Sciences*, vol. 114, no. 23, pp. 5792–5799, 2017.

[9] A. Carbo, R. Hontecillas, B. Kronsteiner, M. Viladomiu, M. Pedragosa, P. Lu, C. W. Philipson, S. Hoops, M. Marathe, S. Eubank *et al.*, "Systems modeling of molecular mechanisms controlling cytokine-driven cd4+ t cell differentiation and phenotype plasticity," *PLoS Computational Biology*, vol. 9, no. 4, p. e1003027, 2013.

[10] H. F. Fumiã and M. L. Martins, "Boolean network model for cancer pathways: predicting carcinogenesis and targeted therapy outcomes," *PloS One*, vol. 8, no. 7, p. e69008, 2013.

[11] S.-H. Cho, S.-M. Park, H.-S. Lee, H.-Y. Lee, and K.-H. Cho, "Attractor landscape analysis of colorectal tumorigenesis and its reversion," *BMC Systems Biology*, vol. 10, pp. 1–13, 2016.

[12] M. Choi, J. Shi, Y. Zhu, R. Yang, and K.-H. Cho, "Network dynamics-based cancer panel stratification for systemic prediction of anticancer drug response," *Nature Communications*, vol. 8, no. 1, p. 1940, 2017.

[13] S. R. Choi, C. Y. Hwang, J. Lee, and K.-H. Cho, "Network analysis identifies regulators of basal-like breast cancer reprogramming and endocrine therapy vulnerability," *Cancer Research*, vol. 82, no. 2, pp. 320–333, 2022.

[14] N. Kim, C. Y. Hwang, T. Kim, H. Kim, and K.-H. Cho, "A cell-fate reprogramming strategy reverses epithelial-to-mesenchymal transition of lung cancer cells while avoiding hybrid states," *Cancer Research*, vol. 83, no. 6, pp. 956–970, 2023.

[15] D. Cheng and H. Qi, "A linear representation of dynamics of boolean networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 10, pp. 2251–2258, 2010.

[16] Y. Zhao, J. Kim, and M. Filippone, "Aggregation algorithm towards large-scale boolean network analysis," *IEEE Transactions on Automatic Control*, vol. 58, no. 8, pp. 1976–1985, 2013.

[17] T. Akutsu, M. Hayashida, W.-K. Ching, and M. K. Ng, "Control of boolean networks: Hardness results and algorithms for tree structured networks," *Journal of Theoretical Biology*, vol. 244, no. 4, pp. 670–679, 2007.

[18] S. Zhu, J. Lu, S.-i. Azuma, and W. X. Zheng, "Strong structural controllability of boolean networks: Polynomial-time criteria, minimal node control, and distributed pinning strategies," *IEEE Transactions on Automatic Control*, vol. 68, no. 9, pp. 5461–5476, 2023.

[19] S. Zhu, J. Cao, L. Lin, J. Lam, and S.-i. Azuma, "Toward stabilizable large-scale boolean networks by controlling the minimal set of nodes," *IEEE Transactions on Automatic Control*, vol. 69, no. 1, pp. 174–188, 2024.

[20] S. Klamt, J. Saez-Rodriguez, J. A. Lindquist, L. Simeoni, and E. D. Gilles, "A methodology for the structural and functional analysis of signaling and regulatory networks," *BMC bioinformatics*, vol. 7, pp. 1–26, 2006.

[21] A. Y. Wen, K. M. Sakamoto, and L. S. Miller, "The role of the transcription factor creb in immune function," *The Journal of Immunology*, vol. 185, no. 11, pp. 6413–6419, 2010.

[22] L. Duan, A. L. Reddi, A. Ghosh, M. Dimri, and H. Band, "The cbl family and other ubiquitin ligases: destructive forces in control of antigen receptor signaling," *Immunity*, vol. 21, no. 1, pp. 7–17, 2004.

[23] K. Yasuda, M. Nagafuku, T. Shima, M. Okada, T. Yagi, T. Yamada, Y. Minaki, A. Kato, S. Tani-Ichi, T. Hamaoka *et al.*, "Cutting edge: Fyn is essential for tyrosine phosphorylation of csk-binding protein/phosphoprotein associated with glycolipid-enriched microdomains in lipid rafts in resting t cells," *The Journal of Immunology*, vol. 169, no. 6, pp. 2813–2817, 2002.

[24] H. Chu, D. Lee, and K.-H. Cho, "Precritical state transition dynamics in the attractor landscape of a molecular interaction network underlying colorectal tumorigenesis," *PloS One*, vol. 10, no. 10, p. e0140172, 2015.

[25] J. Kim, M. Foo, and D. G. Bates, "Computationally efficient modelling of stochastic spatio-temporal dynamics in biomolecular networks," *Scientific Reports*, vol. 8, no. 1, p. 3498, 2018.

[26] M. Cui, C. Cheng, and L. Zhang, "High-throughput proteomics: a methodological mini-review," *Laboratory investigation*, vol. 102, no. 11, pp. 1170–1181, 2022.

[27] D. Raykova, D. Kermpatsou, T. Malmqvist, P. J. Harrison, M. R. Sander, C. Stiller, J. Heldin, M. Leino, S. Ricardo, A. Klemm *et al.*, "A method for boolean analysis of protein interactions at a molecular level," *Nature Communications*, vol. 13, no. 1, p. 4755, 2022.

[28] M. Choi, J. Shi, S. H. Jung, X. Chen, and K.-H. Cho, "Attractor landscape analysis reveals feedback loops in the p53 network that control the cellular response to dna damage," *Science signaling*, vol. 5, no. 251, pp. ra83–ra83, 2012.

[29] J. X. Zhou, A. Samal, A. F. d'Hérouël, N. D. Price, and S. Huang, "Relative stability of network states in boolean network models of gene regulation in development," *Biosystems*, vol. 142, pp. 15–24, 2016.

[30] J. I. Joo, H.-J. Park, and K.-H. Cho, "Normalizing input–output relationships of cancer networks for reversion therapy," *Advanced Science*, vol. 10, no. 24, p. 2207322, 2023.

Jongrae Kim (Senior Member, IEEE) is an Associate Professor at the University of Leeds, Leeds, UK. He received a PhD in Aerospace Engineering from Texas A&M University, College Station, TX, USA in 2002. He was a Post-Doctoral Researcher with the University of California, Santa Barbara, CA, USA in 2002-2003, and a Research Associate with the University of Leicester, Leicester, UK in 2004-2007. He was a Lecturer (Assistant Professor) in Biomedical Engineering/Aerospace Sciences at the University of Glasgow, Glasgow, UK in 2007-2014. His research interests are robustness analysis, guidance navigation & control systems for autonomous mobile systems, dynamics and systems biology.

Woojeong Lee received the B.S. degree in biological science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea. He is currently pursuing the Ph.D. degree in systems biology with the Department of Bio and Brain engineering, KAIST, Daejeon, South Korea. His research interests include the complex network dynamics and Boolean modeling of biological phenomena, especially cancer.

Kwang-Hyun Cho (Senior Member, IEEE) started systems biology by his own idea of combining control engineering and biology in 1999. He has been challenging to develop new control technologies for reverting cancer and aging on the basis of systems biology. He is currently a Professor with the Department of Bio and Brain Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, and the Director of the Laboratory for Systems Biology and Bio-inspired Engineering, KAIST, (http://sbie.kaist.ac.kr). He has authored or coauthored over 222 papers in high-profile international journals. Dr. Cho was a recipient of the IEEE/IEEK Joint Award for Young IT Engineer, the National Young Scientist Award and the National Engineer Award both from the President of Korea, and the Walton Fellow Award from Science Foundation of Ireland. He is an Editor-in-Chief of the IET Systems Biology (IET, London, U.K.) and the Encyclopedia of Systems Biology (Springer, New York).