JKAU: Comp. Sci., **Vol. 4**. 2, pp: 39 - 68 (2015 A.D./ 1436 A.H.)

DOI: 10.4197 / Comp. 4 - 3

Derivation of Reduced Scalar Equations for Synchronous Boolean Networks

Ali Muhammad Ali Rushdi

Department of Electrical and Computer Engineering,

Faculty of Engineering, King Abdulaziz University,

Jeddah, Kingdom of Saudi Arabia

Abstract. This paper studies reduced scalar equations that are used to model synchronous Boolean networks. We distinguish between *individual* minimal equations, which are typically different for the various scalar variables, and a *unified* minimal equation, which is common for all the scalar variables. We review and enhance the existing *ad hoc* method for the derivation of these equations by seeking and utilizing certain *orthogonality* relations among certain successive instances of the same scalar variable. We also present a *novel* general *algorithm* for deriving the reduced scalar equations via powers of either the *function matrix* or the *transition matrix* of the network. The algorithm simplifies considerably in the case of *affine* equations. We present three classical examples to illustrate our techniques, present corrections of previously published results, and demonstrate how the reduced scalar equations can be supplemented by techniques of number theory, Diophantine equations and Boolean equations in making subtle inferences about Boolean networks.

Keywords: Synchronous Boolean networks, Reduced scalar Boolean equations, Orthogonality, Ad hoc derivation, Algorithmic derivation, Affine networks.

1. Introduction

A Synchronous Boolean network consists of a set of n nodes, each of which is either in state 1 (*on*) or state 0 (*off*) at any given time t. Each node is updated at time $(t + 1)$ by inputs from any fixed subset of the set of nodes according to any desired logical rule[12, 14, 16, 19, 33, 34, 36]. All possible trajectories of the network consist of either cycles (loops or attractors) or transient states leading eventually to a cycle. An ideal total description of the network (in which one accounts for all 2^n states) can be achieved by matrix methods[6-9, 33], but can be realized only for small n, and would be unfeasible for large networks. Zhao[52] showed that analysis of Boolean networks involves strong NP-complete problems, which means that this analysis is highly intractable, and that the best algorithms that can ever be devised for it are highly inefficient.

The matrix equations necessary to describe the logic of a given Boolean network can be reduced to higher-order scalar equations which are more transparent to *analyze for cycles*, than the original matrix equations[16, 19, 33, 36]. Linear higher-order reduced scalar equations can be derived from the more rudimentary nonlinear scalar equations. The general form of these reduced scalar equations is:

$$X_i(t + r_i) = X_i(t + s_i). \qquad 1 \leq i \leq n, \qquad (1)$$

Here, we assume that r_i and s_i are the *smallest* integers such that $r_i > s_i$. We call equations (1) the *individual* minimal reduced scalar equations, since they are typically different for the various scalar variables. These equations imply:

$$Any\, possible\, cycle\, length = a\, divisor\, of\, (r_i - s_i),\ 1 \leq i \leq n. \quad (2)$$

$$The\, longest\, possible\, transient\, trajectory = max_i(s_i). \qquad (3)$$

Equations (1) also imply a *unified* minimal reduced scalar equation, which is common for all the scalar variables, namely:

$$X_i(t + r) = X_i(t + s), \qquad 1 \leq i \leq n, \qquad (4)$$

where $s = max_i(s_i), r = s + max_i(r_i - s_i)$. Equation (4) implies

$$Any\, possible\, cycle\, length = a\, divisor\, of\, (r - s), \qquad (5)$$

$$The\, longest\, possible\, transient\, trajectory = s. \qquad (6)$$

Heidel, *et al.*,[19] and Farrow, *et al.*,[16] apparently assumed that r_i and s_i in (1) are the same for all i, and that any single equation of the form (1) can be used as the unified equation (4). The assumption is valid for most of the examples studied by them. However, it is not valid for the small 3-variable example studied by Farrow, *et al.,*[16], for which they asserted *explicitly* that all three nodes have exactly the same reduced scalar equation. The assumption of equality among all $r_i's$ and all $s_i's$ is invalid also for the major 11-variable example studied by Farrow, *et al.,*[16]. In fact, the reduced scalar equation derived therein is neither *minimal* nor valid for all the network variables. That is why their value for the transient period of their example network does not agree with the corresponding value obtained by Cheng, *et al.,*[9] via an *exact* and *correct* matrix method based on a semi-tensor product approach.

This paper strives to contribute new results about the derivation, nature, and utilization of reduced scalar equations, and to reconcile the results of these equations with those of the more powerful (but less insightful) matrix methods. We revise and enhance the existing *ad hoc* method in[16, 19, 36] for the derivation of reduced scalar equations by seeking and utilizing certain *orthogonality* relations among certain successive instances of the same scalar variable. We also present a *novel* general *algorithm* for deriving the reduced scalar equations via powers of the function matrix or the transition matrix of the network. The algorithm simplifies considerably in the case of affine equations. We present three classical examples to illustrate our techniques, offer corrections to previously published results, and illustrate how the reduced scalar equations can be supplemented by techniques of number theory[2, 15], Diophantine equations[2, 3, 15, 47] and Boolean equations[3, 18, 27-31, 39] in making subtle inferences about Boolean networks.

2. Ad hoc derivation

This section discusses and improves the existing ad hoc heuristic[16, 19, 34] for the derivation of minimal reduced scalar equations (both individual and unified). Naturally, there is no algorithmic recipe for such a heuristic procedure, and the best way to present it is to demonstrate it by way of a variety of examples. We utilize various techniques of switching algebra (two-valued Boolean algebra). We adhere to the linear representation (Reed-Müller expansion) of Boolean functions[27, 33]. Therefore, we need to utilize well known properties of the *Exclusive-OR* (*XOR*) binary operator (+), also known as modulo-2 addition, or addition over the simplest finite or Galois Field GF(2)[27, 33]. These properties include well known identities[17, 24] such as:

$$y + y = 0, \tag{7}$$

$$y + 1 = \bar{y}, \tag{8}$$

$$y + 0 = y, \tag{9}$$

$$y + \bar{y} = 1, \tag{10}$$

$$y(1 + y) = y\,\bar{y} = 0, \tag{11}$$

We base our analysis on updating or iterating given equations, and then combining the resulting equations. We simplify our manipulations by seeking and utilizing certain orthogonality relations that exist among successive (albeit not necessarily consecutive) instances of the same scalar variable. We augment the reduced-scalar-equation method by other methods of mathematical reasoning, including number theory[2, 15], Diophantine equations[2, 3, 15, 47], and Boolean equations[3, 18, 27-31]. We demonstrate, contrary to previously published assumptions or assertions, that there is typically no common minimal reduced scalar equation for all the scalar variables, *i.e.*, r_i and s_i in (1) are not necessarily the same for all i. Each variable usually satisfies its own *distinct* individual minimal reduced scalar equation. We also demonstrate that the derivation of an individual minimal reduced scalar equation is achieved not only by *proving* it but also by *disproving* an immediately preceding version of it, when such a version could exist. In fact, one must prove (1) for the smallest integers r_i and s_i, namely r_{im} and s_{im}, where the extra subscript m denotes "*minimal*". However, (1) is valid not only for the particular set of values $r_i = r_{im}$ and $s_i = s_{im}$, but also for the doubly-infinite set of values $r_i = r_{im} + k$ and $s_i = s_{im} + k$, where k is an arbitrary nonnegative integer. If the value of s_i is 0, then s_i is minimal and equals s_{im}. Otherwise, a plausible way to guarantee that the smallest values r_{im} and s_{im} of r_i and s_i were attained is to prove the two relations:

$$X_i(t + r_{im}) = X_i(t + s_{im}), \qquad 1 \le i \le n, \tag{12}$$

$$X_i(t + r_{im} - 1) \ne X_i(t + s_{im} - 1). \qquad 1 \le i \le n, \tag{13}$$

With the proof in (12) and the disproof in (13), one makes sure that (12) is indeed the *true* or *minimal* individual reduced scalar equation.

Despite the useful insight supplied by the reduced scalar equations, they do not provide a total solution like the one offered by the more-powerful matrix methods, and therefore they need be supported by other techniques of mathematical reasoning. The following examples illustrate the Boolean-algebraic techniques necessary for the ad hoc derivation of the reduced scalar equations, present corrections of previously published results, refute purported claims of discrepancies between scalar and matrix methods, and illustrate how the reduced scalar equation can be supplemented by techniques of number theory[2, 15], Diophantine equations[2, 3, 15, 47] and Boolean equations[3, 18, 27-29] in making subtle inferences about Boolean networks.

Example 1

Consider a 3-variables Boolean network with transition equations[16, 36]:

$$x_1(t+1) = x_2(t)\, x_3(t), \tag{14a}$$

$$x_2(t+1) = 1 + x_1(t), \tag{14b}$$

$$x_3(t+1) = x_2(t). \tag{14c}$$

Updating time in (14) by one increment, one obtains:

$$x_1(t+2) = x_2(t+1)\, x_3(t+1) = \big(1 + x_1(t)\big)\, x_2(t). \tag{15}$$

An orthogonality condition emerges from (15) by multiplying its two sides by $x_1(t)$ and utilizing (11), namely

$$x_1(t)x_1(t+2) = 0. \tag{16}$$

Further updating of time in (15) by one increment produces:

$$x_1(t+3) = \big(1 + x_1(t+1)\big)\, x_2(t+1) = \big(1 + x_1(t+1)\big)\big(1 + x_1(t)\big), \tag{17}$$

which is a *scalar equation* for *A* that implies a novel orthogonality condition:

$$x_1(t)x_1(t+3) = 0, \tag{18}$$

and also implies the following orthogonality condition, which is just an update of (16):

$$x_1(t+1)x_1(t+3) = 0. \tag{19}$$

Further updating of time in (17) twice produces (with the invocation of (16) and (11)):

$$x_1(t+5) = \big(1 + x_1(t+3)\big)\big(1 + x_1(t+2)\big)$$

$$= \Big(1 + \big(1 + x_1(t)\big)\big(1 + x_1(t+1)\big)\Big)\big(1 + x_1(t+2)\big)$$

$$= \big(1 + x_1(t+2)\big) + \big(1 + x_1(t)\big)\big(1 + x_1(t+1)\big)\big(1 + x_1(t+2)\big)$$

$= 1 + x_1(t+2) + 1 + x_1(t) + x_1(t+1) + x_1(t+2) + x_1(t)x_1(t+2)$

$+x_1(t)x_1(t+1) + x_1(t+1)x_1(t+2) + x_1(t)x_1(t+1)x_1(t+2),$

which simplifies thanks to (16), (11), (14a), and (14b) to

$$\begin{aligned} x_1(t+5) &= x_1(t) + x_1(t+1) + x_1(t)x_1(t+1) + x_1(t+1)x_1(t+2) \\ &= x_1(t) + x_1(t+1)[1 + x_1(t) + x_1(t+2)] \\ &= x_1(t) + x_2(t)x_3(t)[x_2(t+1) + x_2(t+1)x_2(t)] \\ &= x_1(t) + x_2(t)x_3(t)x_2(t+1)[1 + x_2(t)] = x_1(t) + 0\, x_1(t), \end{aligned} \quad (20)$$

which is an *individual reduced scalar equation for* x_1. It is the *minimal* such equation, since it has no preceding version (Here $s_m = 0$). Likewise, we can prove that x_2 *and* x_3 have similar *scalar equations,* namely:

$$x_2(t+3) = 1 + x_2(t)x_2(t+1), \quad (21)$$

$$x_3(t+3) = 1 + x_3(t)x_3(t+1). \quad (22)$$

and different *individual reduced scalar equations*, namely:

$$x_2(t+6) = x_2(t+1). \quad (23)$$

$$x_3(t+7) = x_3(t+2), \quad (24)$$

The proof ascertains that the preceding candidate equations

$$x_2(t+5) \overset{?}{=} x_2(t). \quad (25)$$

$$x_3(t+6) \overset{?}{=} x_3(t+1), \quad (26)$$

do not hold in general.

Contrary to a claim made in[16], p. 350, the three reduced scalar equations for the variables x_1, x_2, and x_3 (Equations (20), (23), and (24)) are not the same. However, the three equations collectively imply the unified minimal reduced scalar equation:

$$y(t+7) = y(t+2), \quad (27)$$

where y stands for each of x_1, x_2, or x_3. This means that the maximum transient in the state diagram is of length two, and the smallest common multiple of cycle lengths is $7 - 2 = 5$. This allows the possibility of cycles of length equal to divisors of 5 $(i.e. 1\ or\ 5)$. To study the possibility of fixed points (cycle of length 1), add the requirements

$$x_i(t+1) = x_i(t),\ \ i = 1, 2, 3, \tag{28}$$

to equations (14), thereby obtaining:

$$x_1(t) = x_2(t)\, x_3(t), \tag{29a}$$

$$x_2(t) = 1 + x_1(t), \tag{29b}$$

$$x_3(t) = x_2(t). \tag{29c}$$

The system of equations (29) is obviously inconsistent and its solution set is empty, since it leads to $\{x_1(t) = x_2(t)\, x_2(t) = x_2(t) = 1 + x_1(t)\}$, and hence leads to {0 = 1}. The network could have only a single cycle of length 5 since its total number of states is $2^3 = 8$. Since the network has a single cycle of length 5 and one transient of length 2, it must have a second transient of length 1. These results agree with the known state transition diagram of the network[36]. Here, we used very simple reasoning to account for all network states, but this will not suffice when dealing with large complicated networks.

Example 2

A Boolean network considered in[9, 16, 21, 36] has 11 nodes represented by the symbols A through K, with network transition equations:

$$A(t+1) = K(t) + K(t)\, H(t), \tag{30a}$$

$$B(t+1) = A(t) + A(t)\, C(t), \tag{30b}$$

$$C(t+1) = 1 + D(t) + D(t)\, I(t), \tag{30c}$$

$$D(t+1) = J(t)\, K(t), \tag{30d}$$

$$E(t+1) = 1 + C(t) + C(t)\, F(t), \tag{30e}$$

$$F(t+1) = E(t) + E(t)\, G(t), \tag{30f}$$

$$G(t+1) = 1 + B(t)\, E(t), \tag{30g}$$

$$H(t+1) = F(t) + F(t)\, G(t), \tag{30h}$$

$$I(t+1) = H(t) + H(t)\, I(t) = H(t)\, \bar{I}(t), \tag{30i}$$

$$J(t+1) = J(t), \tag{30j}$$

$$K(t+1) = K(t). \tag{30k}$$

Huang and Ingber,[21] have shown that a nontrivial growth attractor exists, assuming that the growth factor (node K) and cell spreading (node J) are both ON, which means that node D is also ON $(K(t) = J(t) = D(t) = 1)$. In this case:

$$A(t+1) = 1 + H(t) = \overline{H}(t) \qquad (30a')$$

$$C(t+1) = I(t) \qquad (30c')$$

Rushdi and Alsogati,[36] showed that the variable I(t) has the orthogonality relations listed in Table 1. Clearly, some of these relations start from the outset, while some start after some delay.

Table 1. Set of Orthogonality Relations for $I(t)$.

Time Distance	First Instance of an Orthogonality Relation
1	$I(t)I(t+1) = 0$
2	------------------
3	$I(t+5)I(t+8) = 0$
4	$I(t+1)I(t+5) = 0$
5	$I(t)I(t+5) = 0$
6	$I(t)I(t+6) = 0$

Rushdi and Alsogati,[36] further proved that

$$I(t+14) = I(t+5) + I(t+4)I(t+7) \neq I(t+5). \qquad (31)$$

$$I(t+15) = I(t+6), \qquad (32)$$

which means that (32) is the individual minimal reduced scalar equation for $I(t)$. This equation is one time-period ahead of that given in[16]. Furthermore, contrary to assumptions made in[16], other variables of the Boolean network do not necessarily have the same individual minimal reduced scalar equation. The unified minimal reduced scalar equation is not

$$y(t+16) = y(t+7), \tag{33}$$

as claimed in[16], but it is

$$y(t+19) = y(t+10), \tag{34}$$

which is the minimal reduced scalar equation for $F(t)$. The maximum delay (transient period) T_t encountered is 10 and not 7. This result is in agreement with that provided by the more elaborate matrix method of Cheng, *et al.*,[9]. Our update of the reduced scalar method refutes the possibility of existence of a discrepancy between scalar and matrix methods.

Example 3

Consider a six-node affine Boolean network (that has linear terms plus constant terms in the Reed-Müller expressions of its next-state functions)[19, 23, 33, 36, 51] with network equations

$$x_1(t+1) = 1 + x_6(t), \tag{35a}$$

$$x_2(t+1) = x_1(t), \tag{35b}$$

$$x_3(t+1) = x_2(t), \tag{35c}$$

$$x_4(t+1) = x_3(t), \tag{35d}$$

$$x_5(t+1) = x_4(t), \tag{35e}$$

$$x_6(t+1) = x_5(t). \tag{35f}$$

Rushdi and Alsogati,[36] proved that

$$x_i(t+6) = 1 + x_i(t) \neq x_i(t), \qquad 1 \leq i \leq 6, \tag{36}$$

and that the individual minimal reduced scalar equation for all six variables and the unified minimal reduced scalar equation are all the same, namely,

$$x_i(t+12) = x_i(t), \qquad 1 \leq i \leq 6. \tag{37}$$

This means that there is no transient state for this network ($n_t = 0$), and hence the length of the maximum transient trajectory is zero ($T_t = 0$), and there are possibly cycles of periods that are divisors of $12\ (1,\ 2,\ 3,\ 4,\ 6,\ and\ 12)$, but thanks to (36), one can negate the possibility of cycles of period six and its divisors $(1,\ 2,\ 3,\ and\ 6)$.

Therefore, the only possible cycle lengths are four and twelve. The total number of states can be expressed as

$$2^6 = 64 = 4n_4 + 12n_{12}, \tag{38a}$$

where n_4 is the number of period-four cycles and n_{12} is the number of period-twelve cycles. Equation (38a) can be rewritten as:

$$n_4 = 16 - 3n_{12}. \tag{38b}$$

Equation (38b) is a special Diophantine equation, *i.e.*, an equation with integer coefficients for which integer solutions are sought[2, 3, 15, 47]. This equation can be solved under the conditions that n_4 *and* n_{12} are nonnegative integers. There are six possible solutions for the pair (n_4, n_{12}), namely $(16,\ 0), (13,\ 1), (10,\ 2), (7,\ 3), (4,\ 4), and\ (1,\ 5)$. The scalar equation technique has no explicit way for distinguishing between these six candidate solutions. The solution $n_4 = 1, n_{12} = 5$ cited by Heidel, *et al.*,[19] has five distinct cycles of period twelve and one cycle of period four. This solution represents the actual network solution[36], but it cannot be singled out by the scalar-equation technique alone. Boolean-equation techniques[3, 13, 19-21, 39] can allow for the solution of states on 4-period cycles, which turn out to be exactly four states constituting (and fitting into) a *single* period-four cycle ($n_4 = 1$). This asserts the existence of five period-twelve cycles ($n_{12} = 5$).

To conclude this section, we note that ad hoc derivation of the reduced scalar equation for a synchronous Boolean network is typically a cumbersome and time-consuming (albeit insightful) task. However, it could be somewhat simplified by adhering to the linear representation (Reed-Müller expansion) of the Boolean functions. Our examples show that this derivation is considerably facilitated by seeking and utilizing orthogonality relations among some successive instances of the same scalar Boolean variable. For some Boolean networks (see, *e.g.*, Example 3), the individual minimal reduced scalar equations are identical for all the Boolean variables. Our examples 1 and 2 demonstrate clearly that this is not always the case. In both examples, each variable has its own individual minimal reduced scalar equation.

There are infinitely many versions of a reduced scalar equation that are all equally suitable for deducing information about the network cycles or attractors. However, it is necessary to identify the most delayed case among the individual minimal reduced scalar equations in order to predict the transient behavior of the network. In some cases (see, *e.g.*, Examples 1 and 2), the individual minimal reduced scalar equation has a preceding version in which the pertinent times could be decremented by a single time period. Here, it is necessary to disprove this preceding version if one is to make sure that the individual reduced equation is truly minimal. In other cases (see, *e.g.*, Example 3), the earlier time instant in the equation is the initial instant and cannot be decremented. Hence, no preceding version of the equation exists, and there is no doubt about the minimality of the equation.

Since the *disproof* required for the equation preceding the individual minimal reduced scalar equation is usually more tedious than the *proof* of that equation itself, the reduced-scalar-equation method does not seem particularly helpful for evaluating transient length. Therefore, one might be content to quickly derive any individual reduced scalar equations, ignoring whether they are the true minimal versions or belated versions, combine them into a unified equation (that is not necessarily minimal), and use this equation to study only the cyclic behavior of the network. This strategy might make the most of reduced scalar equations. There is a paramount interest in the study of cycles in Boolean networks used as models of gene-regulatory network[20]. As a network is trapped in a cycle as soon as one of its states is entered, cycles comprise the states in which the network resides most of the time. It is assumed that cycles in gene-regulatory networks are linked to phenotypes[20]. Therefore, restricting the utility of scalar equations to cyclic behavior does not lead to a significant reduction of their importance.

3. Algorithmic Derivation

The ad hoc procedure of Section 2 is fallible and time consuming. Therefore, an algorithm for deriving the reduced scalar equations is sought. A promising direction is to consider the initial transition equations and a few of their updates as premises and use logic deduction to ferret out the reduced scalar equations from them as consequences. This can be achieved via the Modern Syllogistic Method[4, 5, 17, 27, 35, 40-43], which relies on the computation of the complete sum of a switching function that is obtained via equational forms of the premises[1, 10, 11, 13, 22, 25-27, 32, 37, 38, 44, 45, 48-50].

There is an alternative way for algorithmic derivation of reduced scalar equations that we are going to employ herein, since it does not rely on a faraway area such as logic deduction but rather seeks assistance from the sister area of matrix methods for Boolean networks. This alternative way will also provide insight about the interrelation between scalar and matrix methods for Boolean networks. We employ powers of one of two related matrices, namely, the function matrix $\boldsymbol{A}$ and the transition matrix $\boldsymbol{T}$. The function matrix $\boldsymbol{A}$ is easier to derive and more transparent for scalar variables, while the transition matrix $\boldsymbol{T}$ is easier to check and more transparent for state expressions. The function matrix $\boldsymbol{A}$ has the extra advantage that it has a reduced version that can be used in the special case of affine networks.

3.1. Powers of the Function Matrix

Switching functions used herein are isomorphic to functions over the simplest finite or Galois field GF(2), also known as the binary field or mod-2 field[12, 33]. The field has only two elements: the additive identity (0) and the multiplicative identity (1). The field addition $(+)$ and multiplication $(\times)$ operations are defined by the following axioms:

$$0 + 0 = 1 + 1 = 0, \tag{39a}$$

$$1 + 0 = 0 + 1 = 1, \tag{39b}$$

$$0 \times 0 = 0 \times 1 = 1 \times 0 = 0, \tag{39c}$$

$$1 \times 1 = 1. \tag{39d}$$

Note that the addition $(+)$ operation is a modulo-2 operation that resembles the *exclusive-OR* operation $(\oplus)$ in switching algebra (two-valued Boolean algebra). Any function of n variables over GF(2) is a polynomial of 2^n terms (called a Taylor or a Reed-Müller polynomial[27, 33, 46]), and hence can be represented by a vector of length 2^n, whose elements are the binary coefficients of the 2^n terms in the polynomial. This representation is also called a linear or Boolean-ring representation[27]. In the sequel, we will use the isomorphic representations over GF(2) and of the Boolean ring interchangeably, and in particular our $(+)$ sign is equivalently understood to mean modulo-2 addition or *XOR* operation ($1a$ and $1b$), and our $(\times)$ sign (to be omitted and replaced simply by juxtapositioning) is equivalently understood to mean 1-bit multiplication or *AND* operation ($1c$ and $1d$).

A vector of 2^n terms, arranged in some agreed-upon order, constitutes a basis for all functions of n variables over GF(2) or equivalently all switching functions of n variables, in the sense that any of these functions equals the (scalar) dot product of the vector of binary coefficients representing it with the aforementioned basis vector. There are many different forms of the basis vectors that are typically used implicitly rather than explicitly. The form used herein was proposed by Rushdi and Al-Otaibi,[33] as:

$$\boldsymbol{B}_n = [1 \quad x_1 \quad x_2 \quad x_1x_2 \quad x_3 \quad x_1x_3 \quad x_2x_3x_1x_2x_3 \quad \dots \quad x_n \quad x_1x_n \quad x_2x_n \quad x_1x_2x_n \cdots \quad x_1x_2x_3 \dots x_n]^T. \tag{40}$$

which can be defined by the simple recursive relation:

$$\boldsymbol{B}_n = \begin{bmatrix} \boldsymbol{B}_{n-1} \\ \\ \boldsymbol{B}_{n-1} \wedge x_n \end{bmatrix}, \tag{41a}$$

together with the boundary condition:

$$\boldsymbol{B}_0 = 1. \tag{41b}$$

A restatement of the recursive relation (41) is that none of the logical products (that are elements of $\boldsymbol{B}_n$) is succeeded by products that are subsumed by it. We recall that a product subsumes another if the set of literals of the former is a superset of that of the latter[24]. For example, the product $x_1\bar{x}_2$ subsumes the products 1, x_1, $\bar{x}_2$, and itself.

Now, we review two concepts from Cull[12] and Rushdi and Al-Otaibi[33], concerning a synchronous Boolean network of n nodes. The first concept is that of the *cumulative state vector* $\boldsymbol{X}(t)$. This is a binary column vector of dimension 2^n, which equals the basis vector $\boldsymbol{B}_n$ evaluated at the network state at time instant t. Hence, $\boldsymbol{X}(t)$ has $1's$ in the positions representing any of the products of the elements of the synchronous Boolean network that are in state 1, *i.e.*, the products subsumed by the product depicting the network state. Hence, the number of $1's$ in the vector $\boldsymbol{X}(t)$ is 2^k, where $k = 0, 1, 2, \dots, or\ n$. The second concept is called the *function matrix* $\boldsymbol{A}$. This is a A $2^n \times 2^n$ matrix that has as its rows the vector representations of the 2^n products of the n functions computed by the elements of the synchronous Boolean network. Figure 1 illustrates the construction of the function matrix $\boldsymbol{A}$ for $n = 3$. The function matrix relates two consecutive instances of the vector *X* as:

$$\boldsymbol{X}(t+1) = \boldsymbol{A}\,\boldsymbol{X}(t). \tag{42}$$

By incrementing the time from t to (t+1), one obtains:

$$\boldsymbol{X}(t+2) = \boldsymbol{A}\,\boldsymbol{X}(t+1), \tag{43}$$

which combines with (42) to give

$$\boldsymbol{X}(t+2) = \boldsymbol{A}^2\boldsymbol{X}(t). \tag{44}$$

Similarly, one can generalize (44) to

$$\boldsymbol{X}(t+k) = \boldsymbol{A}^k\boldsymbol{X}(t),\ \ k = 0,1,2,3,\ldots \tag{45}$$

$$\boldsymbol{X}(t+k) = \boldsymbol{A}^{k-m}\boldsymbol{X}(t+m),\ (k-m) = 0,1,2,3,\ldots \tag{46}$$

If the rows of matrices are numbered from 0 to $(2^n - 1)$, then equations (45) and (46) include expressions for each of the scalar variables or singleton products $x_i\ (t+k)$, $(1 \le i \le n)$. Note that $x_i\ (t+k)$ is the Galois-field multiplication of row 2^{i-1} of $\boldsymbol{A}^k$ with column vector $\boldsymbol{X}(t)$, and is also the Galois-field multiplication of row 2^{i-1} of $\boldsymbol{A}^{k-m}$ with column vector $\boldsymbol{X}(t+m)$.

Since the network will ultimately abandon its transients to enter and reside in a cycle (attractor), the matrix powers $\boldsymbol{A}^k$ and $\boldsymbol{A}^m$ become equal for high enough k and m such that $(k-m)$ is the smallest common multiple of all cycle lengths. We call the resulting equation

$$\boldsymbol{A}^k = \boldsymbol{A}^m,\ k > m, \tag{47}$$

for the *smallest* possible values of $k\ and\ m$, the true or minimal *reduced vector equation* of the network for $\boldsymbol{A}$. Once this equation is attained, it remains valid when the indices $k\ and\ m$, are updated equally, *i.e.*, the equations

$$\boldsymbol{A}^{k+j} = \boldsymbol{A}^{m+j},\ k > m, \tag{48}$$

are valid for all non-negative integral values of j.

The scalar variable $x_i\,(t+k)$ is expected to become equal to an earlier instance of it $x_i\ (t+m)$, again for high enough k and m such that $(k-m)$ is the smallest common multiple of all cycle lengths. However, this does not wait for the first occurrence of equality between $\boldsymbol{A}^k$ and $\boldsymbol{A}^m$. In fact, equality of $x_i\,(t+k)$ to $x_i\,(t+m)$ happens when row 2^{i-1} of $\boldsymbol{A}^{k-m}$ becomes row 2^{i-1} of the identity matrix $\boldsymbol{I}$, *i.e.*, a unit row vector of all-0 elements except with 1 at the diagonal element. Equivalently, this equality happens when row 2^{i-1} of $\boldsymbol{A}^k$ becomes equal to row 2^{i-1} of $\boldsymbol{A}^m$, and might happen earlier than the event that $\boldsymbol{A}^k$ and $\boldsymbol{A}^m$ become equal. The equality of $x_i\,(t+k)$ to $x_i\,(t+m)$ corresponds to any updated individual scalar equation, but we are interested in its first occurrence, which signifies the true minimal equation. We will seek this minimal scalar equation by constructing successively increasing powers of $\boldsymbol{A}$ and observing the *first occasion* when any of the rows 1, 2, 4, 8, …, 2^{n-1}, are equal in a certain power $\boldsymbol{A}^k$ and in an earlier power $\boldsymbol{A}^m$, which could possibly be $\boldsymbol{A}^0 = \boldsymbol{I}$.

$\boldsymbol{A}$ =

	1	$x_1(t)$	$x_2(t)$	$x_1(t)\,x_2(t)$	$x_3(t)$	$x_1(t)\,x_3(t)$	$x_2(t)\,x_3(t)$	$x_1(t)x_2(t)\,x_3(t)$
1	1	0	0	0	0	0	0	0
$x_1(t+1)$	Next-State Functions in Linear Form							
$x_2(t+1)$								
$x_1(t+1)x_2(t+1)$	Product of Next-State Functions in Linear Form							
$x_3(t+1)$	Next-State Function in Linear Form							
$x_1(t+1)x_3(t+1)$								
$x_2(t+1)\,x_3(t+1)$	Products of Next-State Functions in Linear Form							
x_1 (t + 1) x_2 (t + 1) x_3 (t + 1)								

Fig. 1. Construction of the function matrix *A* for n = 3.

Example 1 (revisited)

We construct the function matrix $\boldsymbol{A}$, and its powers up to $\boldsymbol{A}^7$ for the network in Example 1. Note that in addition to the expressions in equations (14a)-(14c) which fit into rows 1, 2, and 4 of $\boldsymbol{A}$, we need to compute[33]

$$x_1(t+1)\,x_2(t+1) = x_2(t)\,x_3(t) + x_1(t)x_2(t)\,x_3(t), \tag{14d}$$

$$x_1(t+1)\,x_3(t+1) = x_2(t)\,x_3(t), \tag{14e}$$

$$x_2(t+1)\,x_3(t+1) = x_2(t) + x_1(t)x_2(t), \tag{14f}$$

$$x_1(t+1)x_2(t+1)\,x_3(t+1) = x_2(t)\,x_3(t) + x_1(t)x_2(t)\,x_3(t), \tag{14g}$$

which constitute rows 3, 5, 6, and 7 of $\boldsymbol{A}$ according to the scheme of Fig. 1. The needed powers of $\boldsymbol{A}$ are:

$$\boldsymbol{A} = \begin{array}{|c|c|c|c|c|c|c|c|}
\hline
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline
\end{array} \tag{49a}$$

$$\boldsymbol{A}^2 = \begin{array}{|c|c|c|c|c|c|c|c|}
\hline
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline
1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \hline
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ \hline
1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ \hline
0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \hline
\end{array} \tag{49b}$$

A^3 =

1	0	0	0	0	0	0	0
1	0	1	1	0	0	0	1
1	0	1	1	0	0	0	0
1	1	1	1	0	0	0	0
1	0	0	0	0	0	1	0
1	1	0	0	0	0	1	1
1	0	1	1	0	0	0	1
1	1	1	1	0	0	0	0

(49c)

A^4 =

1	0	0	0	0	0	0	0
1	0	1	1	0	0	0	1
0	1	0	0	0	0	1	1
0	1	0	0	0	0	0	1
1	0	1	1	0	0	0	0
1	0	1	1	0	0	0	1
0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	1

(49d)

A^5 =

1	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	1	1	0	0	0	1
0	0	0	0	0	0	0	1
0	1	0	0	0	0	1	1
0	1	0	0	0	0	0	0
0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1

(49e)

A^6 =

1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	1	1
0	0	1	1	0	0	0	1
0	0	0	0	0	0	1	0
0	0	1	1	0	0	0	0
0	0	0	0	0	0	1	1

(49f)

$$\boldsymbol{A}^7 = \boldsymbol{A}^2. \tag{49g}$$

Equation (49g) is the true or minimal *reduced vector equation* of the network for $\boldsymbol{A}$. It implies the unified minimal reduced scalar equation (27). The minimal reduced scalar equation for x_1 (Eq. (20)) results from equality of row 1 in $\boldsymbol{A}^5$ to row 1 in the identity matrix $\boldsymbol{A}^0 = \boldsymbol{I}$. The minimal reduced scalar equation for x_2 (Eq. (23)) results from equality of row 2 in $\boldsymbol{A}^6$ to row 1 in the matrix $\boldsymbol{A}$ itself. The reduced scalar equation for x_3 (Eq. (24)) results from equality of row 1 in $\boldsymbol{A}^7$ to row 1 in the matrix $\boldsymbol{A}^2$, and hence coincides with the unified reduced scalar equation (27).

3.2 The Affine Case

An affine Boolean network has linear terms plus constant terms in the Reed-Müller expressions of its next-state functions. If the rows of the function matrix $\boldsymbol{A}$ are numbered from 0 to $(2^n - 1)$, then it suffices to retain the rows and columns numbered 2^{i-1} $(0 \leq i \leq n)$. This replaces the 2^n *by* 2^n function matrix $\boldsymbol{A}$ by the *reduced* function matrix $\boldsymbol{A}_r$ of dimensions (n+1) by (n+1). Row 0 of $\boldsymbol{A}_r$ represents the constant term, while each of the other n rows represents an updated scalar variable or a singleton $x_i\,(t+1)$, $(1 \leq i \leq n)$. Likewise, column 0 of $\boldsymbol{A}_r$ represents the constant term, while each of the other n columns represents a scalar variable $x_i\,(t)$, $(1 \leq i \leq n)$. Figure 2 illustrates the construction of the reduced function matrix $\boldsymbol{A}_r$ for $n = 6$.

$\boldsymbol{A}_r$ =

	1	$x_1(t)$	$x_2(t)$	$x_3(t)$	$x_4(t)$	$x_5(t)$	$x_6(t)$
1	1	0	0	0	0	0	0
$x_1(t+1)$	Products of Next-State Functions in Linear Form						
$x_2(t+1)$							
$x_3(t+1)$							
$x_4(t+1)$							
$x_5(t+1)$							
$x_6(t+1)$							

Fig. 2. Construction of the reduced function matrix A_r for n = 6 in the affine case.

Example 3 (revisited)

We construct the reduced function matrix $\boldsymbol{A}_r$, and some of its powers up to $\boldsymbol{A}_r^{12}$ for the affine network in Example 3. The needed powers of $\boldsymbol{A}_r$ are:

$$\boldsymbol{A_r} = \begin{array}{|c|c|c|c|c|c|c|} \hline 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline \end{array} \tag{50a}$$

$$\boldsymbol{A_r}^2 = \begin{array}{|c|c|c|c|c|c|c|} \hline 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline \end{array} \tag{50b}$$

$$\boldsymbol{A_r}^3 = \begin{array}{|c|c|c|c|c|c|c|} \hline 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline \end{array} \tag{50c}$$

$$\boldsymbol{A_r}^5 = \begin{array}{|c|c|c|c|c|c|c|} \hline 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline \end{array} \tag{50d}$$

$A_r{}^6$ =

1	0	0	0	0	0	0
1	1	0	0	0	0	0
1	0	1	0	0	0	0
1	0	0	1	0	0	0
1	0	0	0	1	0	0
1	0	0	0	0	1	0
1	0	0	0	0	0	1

(50e)

$A_r{}^{11}$ =

1	0	0	0	0	0	0
0	0	1	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	1	0	0
0	0	0	0	0	1	0
0	0	0	0	0	0	1
1	1	0	0	0	0	0

(50f)

$$A_r{}^{12} = \boldsymbol{I}, \tag{50g}$$

where $\boldsymbol{I}$ is the identity matrix of dimensions (n+1) by (n+1), *i.e*., 7 by 7. Equation (50g) asserts that the individual minimal reduced scalar equations for all six variables and the unified minimal reduced scalar equation are all the same (Eq. (36)).

3.3 Powers of the Transition Matrix

To introduce the concept of the transition matrix $\boldsymbol{T}$, we note that it is used to update the exact state vector $\boldsymbol{Y}(t)$ via

$$\boldsymbol{Y}(t+1) = \boldsymbol{T}\,\boldsymbol{Y}(t). \tag{51}$$

The exact state vector $\boldsymbol{Y}(t)$ is a binary column vector of dimension 2^n, whose elements are all 0 except one element of value 1. This particular element of value 1 corresponds to the position where the product depicting the network state occurs in the basis vector $\boldsymbol{B}_n$ given by (40) or (41). The Transition Matrix $\boldsymbol{T}$ is a $2^n \times 2^n$ matrix having exactly a single 1 in each column and 0 elements otherwise, and it represents the state transition diagram of the network. Figure 3 demonstrates the construction of $\boldsymbol{T}$ for n = 3. All powers of $\boldsymbol{T}$ inherit its characteristic of having exactly a single 1 in each column and 0 elements otherwise, and hence their correctness can be easily checked. The transition matrix $\boldsymbol{T}$ and the function matrix $\boldsymbol{A}$ are related via the state matrix $\boldsymbol{S}_n$ by the following similarity transformations[12, 33, 46]:

$$\boldsymbol{A}\,\boldsymbol{S}_n = \boldsymbol{S}_n\,\boldsymbol{T}\,, \tag{52}$$

$$\boldsymbol{A} = \boldsymbol{S}_n\,\boldsymbol{T}\,\boldsymbol{S}_n^{\;-1} = \boldsymbol{S}_n\,\boldsymbol{T}\,\boldsymbol{S}_n, \tag{53}$$

$$\boldsymbol{T} = \boldsymbol{S}_n^{\;-1}\boldsymbol{A}\,\boldsymbol{S}_n = \boldsymbol{S}_n\,\boldsymbol{A}\,\boldsymbol{S}_n. \tag{54}$$

The matrix $[S_n]$ is a self-inverse given recursively by[33, 46]:

$$\boldsymbol{S}_0 = \mathbf{1}, \tag{55a}$$

$$\boldsymbol{S}_n = \begin{bmatrix} \boldsymbol{S}_{n-1} & \boldsymbol{S}_{n-1} \\ \mathbf{0}_{n-1} & \boldsymbol{S}_{n-1} \end{bmatrix}, \qquad n > 0, \tag{55b}$$

where $\mathbf{0}_{n-1}$ is the zero matrix of dimensions $2^{n-1} \times 2^{n-1}$. The similarity in (53) and (54) between $\boldsymbol{T}$ and $\boldsymbol{A}$ extends to similarity between their powers $\boldsymbol{T}^k$ and $\boldsymbol{A}^k$, where k is any natural integer $(k = 0, 1, 2, \cdots)$, *i.e.*

$$\boldsymbol{T}^k = \boldsymbol{S}_n\,\boldsymbol{A}^k\,\boldsymbol{S}_n, \tag{56}$$

$$\boldsymbol{A}^k = \boldsymbol{S}_n\,\boldsymbol{T}^k\,\boldsymbol{S}_n. \tag{57}$$

The transition matrix $\boldsymbol{T}$ can be obtained from the function matrix $\boldsymbol{A}$ via (54)[12, 33, 36]. It can be also obtained directly from the initial network transition equations via the powerful semi-tensor product (STP) methodology[9], though the STP community calls it the structure matrix $\boldsymbol{L}$, and represents it via a different basis vector.

When the network ultimately abandons its transients to enter and reside in a cycle (attractor), the matrix powers $\boldsymbol{T}^k$ and $\boldsymbol{T}^m$ become equal for high enough k and m such that $(k - m)$ is the smallest common multiple of all cycle lengths. In fact, the similarity relation (56) can be used to convert the true or minimal *reduced vector equation* of the network for $\boldsymbol{A}$ to a similar one for $\boldsymbol{T}$, namely

$$\boldsymbol{T}^k = \boldsymbol{T}^m,\; k > m. \tag{58}$$

Once this minimal *reduced vector equation* of the network for $\boldsymbol{T}$ is attained, it remains valid for any similar updates of the indices k *and* m, *i.e.*, the equations

$$\boldsymbol{T}^{k+j} = \boldsymbol{T}^{m+j},\; k > m, \tag{59}$$

are valid for all non-negative integral values of j.

When the transition matrix $\boldsymbol{T}$ is obtained independently from the function matrix $\boldsymbol{A}$ (*e.g.*, via the semi-tensor product (STP) methodology[9]), then its *minimal reduced vector equation* can lead to the *unified minimal* reduced scalar equations. However, it is of limited help in observing the minimal individual reduced scalar equations, since one should compute for two different powers of $\boldsymbol{T}$ the modulo-2 summation of the rows for which the pertinent variable appears un-complemented. Therefore, the transition matrix $\boldsymbol{T}$ might be used for just verifying (rather than obtaining) these equations.

$\boldsymbol{T}$ =

	$\bar{x}_1\bar{x}_2\bar{x}_3$	$x_1\bar{x}_2\bar{x}_3$	$\bar{x}_1x_2\bar{x}_3$	$x_1x_2\bar{x}_3$	$\bar{x}_1\bar{x}_2x_3$	$x_1\bar{x}_2x_3$	$\bar{x}_1x_2x_3$	$x_1x_2x_3$
$\bar{x}_1(t+1)\bar{x}_2(t+1)\bar{x}_3(t+1)$	All elements of column j of $\boldsymbol{T}$ are zeroes with the exception of a single element t_{ij} which is 1, indicating that state i is the unique next state of current state j.							
$x_1(t+1)\bar{x}_2(t+1)\bar{x}_3(t+1)$								
$\bar{x}_1(t+1)x_2(t+1)\bar{x}_3(t+1)$								
$x_1(t+1)x_2(t+1)\bar{x}_3(t+1)$								
$\bar{x}_1(t+1)\bar{x}_2(t+1)x_3(t+1)$								
$x_1(t+1)\bar{x}_2(t+1)x_3(t+1)$								
$\bar{x}_1(t+1)x_2(t+1)x_3(t+1)$								
$x_1(t+1)x_2(t+1)x_3(t+1)$								

Fig. 3. Construction of the transition matrix $\boldsymbol{T}$ for n = 3, where $\bar{x}_i$ stands for $\bar{x}_i(t)$.

Example 1 (revisited)

We construct the transition matrix $\boldsymbol{T}$ for the network in Example 1 via (54) and (55)[33] and construct also its powers up to $\boldsymbol{T}^7$.

$$\boldsymbol{T} = \begin{array}{|c|c|c|c|c|c|c|c|}
\hline
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline
\end{array} \qquad (60a)$$

$$\boldsymbol{T}^2 = \begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&0&0&0&0&0&0&1\\\hline 0&0&0&0&0&0&0&0\\\hline 0&1&0&1&0&1&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&0&0&0&0&0&1&0\\\hline 1&0&0&0&1&0&0&0\\\hline 0&0&1&0&0&0&0&0\\\hline \end{array}$$

(60b)

$$\boldsymbol{T}^4 = \begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&0&1&0&0&0&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&0&0&0&0&0&1&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 1&0&0&0&1&0&0&0\\\hline 0&0&0&0&0&0&0&1\\\hline 0&1&0&1&0&1&0&0\\\hline \end{array}$$

(60c)

$$\boldsymbol{T}^5 = \begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&1&0&0&1&0&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&0&1&0&0&0&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&1&0&1&0&1&0&0\\\hline 0&0&0&0&0&0&1&0\\\hline 0&0&0&0&0&0&0&1\\\hline \end{array}$$

(60d)

$$\boldsymbol{T}^6 = \begin{array}{|c|c|c|c|c|c|c|c|}\hline 0&1&0&1&0&1&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 1&0&0&0&1&0&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&0&0&0&0&0&0&0\\\hline 0&0&0&0&0&0&0&1\\\hline 0&0&1&0&0&0&0&0\\\hline 0&0&0&0&0&0&1&0\\\hline \end{array}$$

(60e)

$$\boldsymbol{T}^7 = \boldsymbol{T}^2 . \tag{60f}$$

Equation (60f) is the true or minimal *reduced vector equation* of the network for $\boldsymbol{T}$. It implies the unified reduced scalar equation (27). The modulo-2 summation of rows 1, 3, 5, and 7 in each of $\boldsymbol{T}^5$ and the identity matrix $\boldsymbol{T}^0$ is equal to

0	**1**	0	**1**	0	**1**	0	**1**

(61)

Hence, the individual minimal reduced scalar equation for x_1 (Eq. (20)) results because**:**

$$x_1(t+5) = x_1(t+5)\bar{x}_2(t+5)\bar{x}_3(t+5) + x_1(t+5)x_2(t+5)\bar{x}_3(t+5) + x_1(t+5)\bar{x}_2(t+5)x_3(t+5) + x_1(t+5)x_2(t+5)x_3(t+5)$$
$$= x_1(t)\bar{x}_2(t)\bar{x}_3(t) + x_1(t)x_2(t)\bar{x}_3(t) + x_1(t)\bar{x}_2(t)x_3(t) + x_1(t)x_2(t)x_3(t) = x_1(t). \tag{62}$$

The modulo-2 summation of rows 2, 3, 6, and 7 in each of $\boldsymbol{T}^6$ and $\boldsymbol{T}$ is equal to

1	0	**1**	0	**1**	0	**1**	0

(63)

Hence, the individual minimal reduced scalar equation for x_2 (Eq. (23)) results because:

$$x_2(t+6) = \bar{x}_1(t+6)x_2(t+6)\bar{x}_3(t+6) + x_1(t+6)x_2(t+6)\bar{x}_3(t+6) + \bar{x}_1(t+6)x_2(t+6)x_3(t+6) + x_1(t+6)x_2(t+6)x_3(t+6)$$
$$= \bar{x}_1(t+1)x_2(t+1)\bar{x}_3(t+1) + x_1(t+1)x_2(t+1)\bar{x}_3(t+1) + \bar{x}_1(t+1)x_2(t+1)x_3(t+1) + x_1(t+1)x_2(t+1)x_3(t+1) = x_2(t+1). \tag{64}$$

The individual minimal reduced scalar equation for x_3 (Eq. (24)) can result from equality of the modulo-2 summation of rows 4, 5, 6, and 7 in $\boldsymbol{T}^7$ to the corresponding quantity in the matrix $\boldsymbol{T}^2$, but it results immediately from the equality of $\boldsymbol{T}^7$ to $\boldsymbol{T}^2$.

4. Conclusions

We achieved a better understanding of the nature of reduced scalar equations, and distinguished between the individual minimal equations, which are typically different for the various scalar variables, and a unified minimal equation, which is common for all the scalar variables. We also enhanced the existing ad hoc method for the derivation of reduced scalar equations by seeking and utilizing certain *orthogonality* relations among certain successive instances of the same scalar variable. We demonstrated Boolean-algebraic techniques for deriving these equations, presented other mathematical tools for utilizing them, and finally reconciled the scalar methods with the more encompassing but more complex and less insightful matrix methods. In particular, we demonstrated, contrary to previously published assumptions or assertions, that there is typically no common minimal reduced scalar equation for all the scalar variables. We also demonstrated that the derivation of an individual minimal reduced scalar equation necessitates validating it for its earliest occurrence and hence is achieved not only by *proving* it but also by *disproving* an immediately preceding version of it when such a version happens to exist.

We supplemented the existing ad hoc derivation of reduced scalar equations by a novel method for the algorithmic derivation of these equations. This algorithmic derivation is based on matrix methods for Boolean networks, and hence it provides insight about the interrelation between scalar and matrix methods for Boolean networks. In this derivation, we employed powers of either the function matrix $\boldsymbol{A}$ or the transition matrix $\boldsymbol{T}$. The function matrix $\boldsymbol{A}$ is easier to derive and more transparent for scalar equations, while the transition matrix $\boldsymbol{T}$ is easier to check and more transparent for state equations. The function matrix $\boldsymbol{A}$ has the extra advantage that it has a reduced version that can be used in the special case of affine networks.

References

[1] **Alexe, G., Alexea, S., Crama, Y., Foldes, S., Hammer, P. L.** and **Simeone, B.,** Consensus algorithms for the generation of all maximal bicliques, *Discrete Applied Mathematics*, **145** (1): 11-21, (2004).

[2] **Andreescu, T.** and **Andrica, D.,** Diophantine Equations, Chapter 8, pp: 145-165, in *Number Theory: Structures, Examples and Problems*, Birkhäuser, Boston, USA, (2009).

[3] **Andreescu, T., Andrica, D.** and **Cucurezeanu, I.,** *An Introduction to Diophantine Equations: A Problem-Based Approach*, Birkhäuser, New York, USA, (2010).

[4] **Blake, A.,** *Canonical Expressions in Boolean Algebra*, Ph. D. Dissertation, Department of Mathematics, University of Chicago, Chicago, Illinois, USA, (1937).

[5] **Brown, F. M.,** *Boolean Reasoning: The Logic of Boolean Equations*. Kluwer Academic Publishers, Boston, USA, (1990), 2nd Ed., Dover Publications, Mineola, NY, USA, (2003).

[6] **Cheng, D.,** Input-state approach to Boolean networks. *IEEE Transactions on Neural Networks*, **20** (3): 512–521, (2009).

[7] **Cheng, D.,** and **H. Qi,** State–space analysis of Boolean networks. *IEEE Transactions on Neural Networks*, **21** (4): 584-594, (2010).

[8] **Cheng, D.,** and **H. Qi,** A linear representation of dynamics of Boolean networks. *IEEE Transactions on Automatic Control*, **55** (10): 2251–2258, (2010).

[9] **Cheng, D., Qi, H.** and **Li**, **Z. Q.** *Analysis and Control of Boolean Networks: A Semi-Tensor Product Approach*, Springer-Verlag, London, (2011).

[10] **Coudert, O.** and **Madre, J. C.,** A new method to compute prime and essential prime implicants of Boolean functions, *Advanced Research in VLSI and Parallel Systems, Proceedings of the 1992 Brown/MIT Conference*, T. Knight and J. Savage (Editors): 113-128, (1992).

[11] **Crama, Y.** and **Hammer**, **P. L.** *Boolean Functions: Theory, Algorithms, and Applications*, Cambridge University Press, Cambridge, United Kingdom, (2011).

[12] **Cull, P.,** Linear analysis of switching nets, *Kybernetik*, **8** (1): 31-39, (1971).

[13] **Cutler, R. B., Kinoshita, K.** and **Muroga, S.,** *Exposition of Tison's Method to Derive all Prime Implicants and all Irredundant Disjunctive Forms for a Given Switching Function*, Report No. UIUCDCS-R-79-993, Department of Computer Science, University of Illinois, Urbana, Illinois, USA, (1979).

[14] **Darby, M.** and **Mysak, L.,** A Boolean delay equation model of an inter-decadal Arctic climate cycle, *Climate Dynamics*, **8**: 241-246, (1993).

[15] **Everest, G.** and **Ward, T.,** Diophantine Equations, Chapter 2 (pp. 43-58) in *An Introduction to Number Theory*, Springer-Verlag, London, UK, (2005).

[16] **Farrow, C., Heidel, J., Maloney, H.** and **Rogers, J.,** Scalar equations for synchronous Boolean networks with biological applications. *IEEE Transactions on Neural Networks*, **15** (2): 348–354, (2004).

[17] **Gregg, J. R.,** *Ones and Zeros: Understanding Boolean Algebra, Digital Circuits, and the Logic of Sets*. IEEE Press, New York, NY, USA, (1998).

[18] **Hammer, P. L.** and **Rudeanu, S.,** *Boolean Methods in Operations Research and Related Areas*. Springer-Verlag, Berlin, Germany, (1968).

[19] **Heidel, J., Maloney, J., Farrow, J.** and **Rogers, J.,** Finding cycles in synchronous Boolean networks with applications to biochemical systems, *International Journal Bifurcation and Chaos*, **13** (3): 535-552, (2003).

[20] **Hopfensitz, M., Mussel, C., Maucker, M.** and **Kestler, H. A.,** Attractors in Boolean networks: A tutorial, *Computational Statistics*, **28** (1): 19-36, (2013).

[21] **Huang, S.** and **Ingber, I.,** Shape-dependent control of cell growth, differentiation, and apoptosis: Switching between attractors in cell regulatory networks, *Experimental Cell Research*, **261**: 91-103, (2000).

[22] **Hwa, H. R.,** A method for generating prime implicants of a Boolean expression, *IEEE Transactions on Computers,* **C-23** (6) : 637-641, (1974).

[23] **Milligan, D. K.** and **Wilson, M. J. D.,** The behavior of affine Boolean sequential networks. *Connectivity Science*, **5** (2): 153-167, (1993).

[24] **Muroga, S.,** *Logic Design and Switching Theory*, Wiley, New York, NY, USA, (1979).

[25] **Pawlak, Z.** and **Skowron, A.,** Rough sets and Boolean reasoning, *Information Sciences*, **177** (1): 41-73, (2007).

[26] **Reusch, B.,** Generation of prime implicants from subfunctions and a unifying approach to the covering problem, *IEEE Transaction on Computers*, **C-24** (9): 924-930, (1975).

[27] **Rudeanu, S.,** *Boolean Functions and Equations*, North-Holland Publishing Company and American Elsevier, Amsterdam, the Netherlands, (1974).

[28] **Rushdi, A. M.,** Using variable-entered Karnaugh maps to solve Boolean equations, *International Journal of Computer Mathematics*, **78** (1): 23-38, (2001).

[29] **Rushdi, A. M.,** Prime-implicant extraction with the aid of the variable-entered Karnaugh map, *Umm Al-Qura University Journal : Science, Medicine and Engineering,* **13** (1): 53-74, (2001).

[30] **Rushdi, A. M.,** Efficient solution of Boolean equations using variable-entered Karnaugh maps, *Journal of King Abdulaziz University: Engineering Sciences*, **15** (1): 105-121, (2004).

[31] **Rushdi, A. M. A.** and **Albarakati, H. M.,** The inverse problem for Boolean equations, *Journal of Computer Science*, **8** (12): 2098-2105, (2012).

[32] **Rushdi, A. M. A.,** and **Albarakati, H. M.**, Prominent classes of the most general subsumptive solutions of Boolean equations, *Information Sciences*, **281**: 53-65, (2014).

[33] **Rushdi, A. M.** and **Al-Otaibi, S. O.,** On the linear analysis of synchronous switching networks. *Journal of King Abdulaziz University: Engineering Sciences*, **18** (2): 43-72, (2007).

[34] **Rushdi, A. M.** and **Al-Otaibi, S. O.,** On limitations of using scalar equations for analyzing synchronous Boolean networks. *Journal of King Abdulaziz University: Engineering Sciences*, **19** (2): 41-49, (2008).

[35] **Rushdi, A. M.** and **Al-Shehri, A. S.,** Logical reasoning and its supporting role in the service of security and justice, *Journal of Security Studies,* 11(22): 115-153, (2002).

[36] **Rushdi, A. M. A. and Alsogati, A. A., On reduced scalar equations for synchronous Boolean networks, Journal of Mathematics and Statistics, 9 (3): 262-276, (2013).**

[37] **Rushdi, A. M. and Al-Yahya, H. A., A Boolean minimization procedure using the variable-entered Karnaugh map and the generalized consensus concept, International Journal of Electronics, 87 (7): 769-794, (2000).**

[38] **Rushdi, A. M. and Al-Yahya, H. A., Derivation of the complete sum of a switching function with the aid of the variable-entered Karnaugh map, Journal of King Saud University: Engineering Sciences,13 (2): 239-269, (2001).**

[39] **Rushdi, A. M.** and **Amashah, M. H.,** Using variable–entered Karnaugh maps to produce compact parametric solutions of Boolean equations, *International Journal of Computer Mathematics*, **88** (15): 3136-3149, (2011).

[40] **Rushdi, A. M.** and **Ba-Rukab, O. M.,** Some Engineering Applications of the modern syllogistic method, SEC7 Paper 226, *Proceedings of the 7th Saudi Engineering Conference (SEC7*), Riyadh, Saudi Arabia, **4**: 389-401, (2007).

[41] **Rushdi, A. M.,** and **Ba-Rukab, O. M.,** The modern syllogistic method as a tool for engineering problem solving, *Journal of Qassim University: Engineering and Computer Sciences,* **1** (1): 57-70, (2008).

[42] **Rushdi, A. M.** and **Ba-Rukab, O. M.,** Powerful features of the modern syllogistic method of propositional logic, *Journal of Mathematics and Statistics,* **4** (3): 186-193, (2008).

[43] **Rushdi, A. M.** and **Ba-Rukab, O. M.,** An exposition of the modern syllogistic method of propositional logic, *Umm Al-Qura University Journal: Engineering and Architecture,* **1** (1): 17-49, (2009).

[44] **Rushdi, A. M. A.** and **Ba-Rukab, O. M.,** Switching-algebraic analysis of relational databases, *Journal of Mathematics and Statistics,* **10** (2): 231-243, (2014).

[45] **Rushdi, A. M.** and **Baz, A. O.,** Computer-assisted resolution of engineering ethical dilemmas, SEC7 Paper 147, *Proceedings of the 7th Saudi Engineering Conference (SEC7)*, Riyadh, Saudi Arabia, **5**: 409-418, (2007).

[46] **Rushdi, A. M. A.** and **Ghaleb, F. A.,** On self-inverse binary matrices over the binary Galois field, *Journal of Mathematics and Statistic*s, **9** (3): 238-248, (2013).

[47] **Schroeder, M.,** Diophantine Equations, Chapter 7, pp: 119-137, in *Number Theory in Science and Communications*, Fifth Edition, Springer-Verlag, Berlin, Germany, (2009).

[48] **Slagle, J. R., Chang**, **C. L.** and **Lee, R. C. T.,** A new algorithm for generating prime implicants, *IEEE Transactions on Computers*, **C-19** (4): 304–310, (1970).

[49] **Ślęzak, D.,** Association reducts: Boolean representation, in **G. Wang,** ***et al***. (Editors), *Rough Sets and Knowledge Technology*, RSKT, LNAI 4062, Springer, Berlin-Heidelberg, Germany: 305-312, (2006).

[50] **Tison, P.,** Generalization *of consensus* theory and application to the minimization of Boolean functions, *IEEE Transactions on Electronic Computers*, **EC-16** (4): 446-456, (1967), with Comments, **Cutler, R. B**., and **Muroga, S**., *ibid.,* **28** (7): 542-543, (1979).

[51] **Wilson, M. J. D.** and **Milligan, D. K.,** Cyclic behavior of autonomous synchronous Boolean networks: Some theorems and conjectures, *Connectivity Science*, **4** (2): 143-154, (1992).

[52] **Zhao, Q.,** A remark on "Scalar equations for synchronous Boolean networks with biological applications," *IEEE Transactions on Neural Networks*, **16** (6): 1715–1716, (2005).

اشتقاق المعادلات المقياسية المختزلة للشبكات البولانية المتزامنة

علي محمد علي رشدي

قسم الهندسة الكهربائية وهندسة الحاسبات، كلية الهندسة،

جامعة الملك عبدالعزيز، جدة، المملكة العربية السعودية

arushdi@kau.edu.sa

المستخلص. تدرس ورقة البحث هذه المعادلات المقياسية المختزلة التي تستخدم لنمذجة الشبكات البولانية المتزامنة. نميز بين المعادلات الفردية الأصغرية، التي عادة ما تكون مختلفة لمختلف المتغيرات المقياسية، ومعادلة أصغرية موحدة تكون مشتركة بين جميع المتغيرات المقياسية. نراجع ونعزز الطريقة الموجودة حاليًا لاشتقاق مثل هذه المعادلات وهي طريقة مبنية على المعالجة المخصصة لكل مسألة على حدة، وذلك من خلال تقصي واستغلال علاقات تعامدية معينة بين قيم متتالية معينة لنفس المتغير المقياسي. نقدم أيضًا خوارزمية عامة جديدة لاشتقاق المعادلات المقياسية المختزلة تستخدم قوى مصفوفة الدوال أو مصفوفة الانتقال للشبكة. وتتبسط هذه الخوارزمية إلى حد كبير في حالة المعادلات الخطية. نقدم ثلاثة أمثلة نمطية لتوضيح الخوارزمية المقترحة، ولتقديم تصحيحات لبعض النتائج التي نشرت سابقًا، وإظهار كيف يمكن أن تُستكمَل طريقة المعادلات المقياسية المختزلة بطرائق نظرية الأعداد، ومعادلات ديوفانتاين، والمعادلات البولانية، وذلك لعمل استدلالات متقنة حول الشبكات البولانية.

الكلمات المفتاحية: الشبكات البولانية المتزامنة، المعادلات البولانية المقياسية المختزلة، التعامدية، الاشتقاق المخصص لكل مسألة على حدة، الاشتقاق الخوارزمي، الشبكات الخطية.