Output Selection and Observer Design for Boolean Control Networks: A Sub-Optimal Polynomial-Complexity Algorithm

Eyal Weiss and Michael Margaliot

Abstract—We derive a new graph-theoretic sufficient condition for observability of a Boolean control network (BCN). We describe two algorithms that are based on this condition. The first selects a set of nodes so that observing this set makes the BCN observable. The second algorithm builds an observer for the observable BCN. Both algorithms are sub-optimal, as they are based on a sufficient but not necessary condition for observability. Yet their time-complexity is linear in the length of the description of the BCN, rendering them feasible for large-scale BCNs. We discuss how these results can be used to provide a sub-optimal yet polynomial-complexity algorithm for the minimal observability problem in BCNs. Some of the theoretical results are demonstrated using BCN models of mammalian cell cycle control, and T-cell receptor kinetics.

Index Terms—Boolean control networks and logic networks, observers for nonlinear systems, systems biology.

I. INTRODUCTION

B OOLEAN networks (BNs) are discrete-time dynamical systems with Boolean state-variables (SVs) and Boolean update functions. BNs have found wide applications in modeling and analysis of dynamical systems. They have been used to capture the existence and directions of links in complex systems (see [30]), to model social networks (see [14], [21]), and the spread of epidemics [18].

Recently, BNs have been extensively used in systems biology (see [16], [26], [32]). A typical example is modeling gene regulation networks using BNs (see [1], [19]). Here the state of each gene, that may be expressed or not, is modeled using a Boolean SV. The interactions between the genes (e.g., through the effect of the proteins that they encode on the promoter regions of other genes) determine the Boolean update function for each SV.

BNs with (Boolean) control inputs are called Boolean control networks (BCNs). Cheng [4] used the semi-tensor product (STP) of matrices to represent the dynamics of a BCN in a form similar to that of a discrete-time linear control system. This led to the analysis of many control-theoretic problems for BCNs, including controllability [7], [22] observability [7], [10], disturbance decoupling [5], stabilization by state feedback [25] and optimal control [11], [24], [37]. However, the STP representation is exponential in the number of SVs, and thus it cannot be used in general to develop efficient algorithms for addressing these problems.

In many real-world systems it is not possible to measure all the SVs, and state observers are needed to reconstruct the entire state of the system based on a time sequence of what can be measured, i.e., the system outputs. An estimate of the entire state is useful in many applications, as it greatly assists in monitoring and control. For example, in bio-reactors one may use sensors to measure variables such as dissolved oxygen, pH and temperature, yet key variables such as biomass and product concentrations can be much more difficult (and costly) to measure [15]. A typical application of state-observers is to integrate their estimates in full-state feedback controllers (see [31]).

Observers can be designed if the BCN is observable [10]. There are several different definitions for observability in [7], [10], [23], and [39]. See also the notion of state observable graphs in [17], and some relations to set controllability [6]. A comparison between these definitions is given in [35].

A non observable system can be made observable by placing additional sensors that measure more (functions of the) SVs. Of course, this may be costly in terms of resources, so a natural question is: find the minimal number of measurements to add so that the resulting system is observable. This minimal observability problem is also interesting theoretically, as its solution means identifying the (functions of) SVs that provide the maximal information on the entire state of the system [27]. Indeed, minimal observability problems have recently attracted considerable interest. Examples include monitoring complex services by minimal logging [2], the optimal placement of phasor measurement units in power systems (see [28]), and the minimal sparse observability problem addressed in [29].
Testing observability of BCNs is NP-hard in the number of SVs [23]. This means that, unless P=NP, it is computationally intractable to determine whether a large BCN is observable. This implies that the minimal observability problem in BCNs is also NP-hard, since it must entail analyzing observability.

It is thus not surprising that many observers for BCNs have exponential complexity. These include the Shift-Register observer, the Multiple States observer [10] and the Luenberger-like observer [38]. This implies that these algorithms cannot be used in large-scale networks.

This letter is motivated by recent work on the minimal observability problem for a special class of BNs called conjunctive Boolean networks (CBNs). In a CBN, every update function is comprised of only AND operations. These include the Shift-Register computation in CBNs; and (2) designing observers for observable CBNs.

Here, we show that a similar graph-theoretic approach provides a sufficient (but not necessary) condition for observability of (general) BCNs. This induces algorithms for solving Problems (1) and (2) above. Now the algorithms are not optimal anymore, i.e., they may add more observations than the minimal number that is indeed required. Nevertheless, they retain their polynomial-time complexity implying that they are feasible for large-scale BCNs. Of course, in the particular case where all the update functions are AND gates these algorithms become optimal.

The main contributions with respect to [33] are that we consider here general networks and with control inputs and not only conjunctive Boolean networks. This also requires introducing a new structure, namely, the reduced dependency graph $G_r$.

We note in passing that the special structure of CBNs makes them amenable to analysis (see [3], [12], [13], [34]) and we believe that more results from this field can be extended to handle general BNs.

II. PRELIMINARIES

Let $S := [0, 1]$. For two integers $i, j$ let $[i, j] := [i, i + 1, \ldots, j]$. A CBN with $n$ SVs, $p$ inputs and $m$ outputs can be represented by the following equations:

$$X_i(k+1) = f_i(X(k), U(k)), \quad \forall i \in [1, n],$$

$$Y_j(k) = h_j(X(k)), \quad \forall j \in [1, m],$$

(1)

where the state vector at time $k$ is denoted by $X(k) := [X_1(k) \ldots X_n(k)] \in S^n$, the input vector by $U(k) := [U_1(k) \ldots U_p(k)] \in S^p$, the output vector by $Y(k) := [Y_1(k) \ldots Y_m(k)] \in S^m$ and $f_i, h_j$ are Boolean functions.

In principle, an update function may include an SV (or control input) that has no effect on the function, e.g., $f_1(X_1(k), X_2(k)) = X_1(k) \lor (X_2(k) \land \bar{X}_2(k))$. We assume that such arguments have been removed, in other words, if $X_i(k)$ is an argument of $f_j$ then there exists an assignment of $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n, U$ such that

$$f_j(X_1, \ldots, X_{i-1}, 0, X_{i+1}, \ldots, X_n, U) \neq f_j(X_1, \ldots, X_{i-1}, 1, X_{i+1}, \ldots, X_n, U).$$

Similarly, a control input appears in an update function only if it really affects the function.

If for some $i$ there exists an output $Y_j(k) = X_i(k)$ then we say that the SV $X_i$ is directly observable or directly measurable.

Let $G = (V, E)$ be a directed graph (digraph), with $V$ the set of vertices, and $E$ the set of directed edges (arcs). Let $e_i \rightarrow_j$ (or $(v_i \rightarrow v_j)$) denote the arc from $v_i$ to $v_j$. When such an arc exists, we say that $v_j$ is an in-neighbor of $v_i$, and $v_i$ as an out-neighbor of $v_i$. The set of in-neighbors [out-neighbors] of $v_i$ is denoted by $N_{in}(v_i)$ [$N_{out}(v_i)$]. The in-degree [out-degree] of $v_i$ is $|N_{in}(v_i)|$ [$|N_{out}(v_i)|$]. A source [sink] is a node with in-degree [out-degree] zero.

For $v_i, v_j \in V$ a walk from $v_i$ to $v_j$, denoted $w_{ij}$, is a sequence: $v_i v_1, v_1 v_2, \ldots, v_{q-1} v_q = v_j$, with $v_i, v_q = v_j$, and $e_i \rightarrow_{v_i} e_{i+1} \in E$ for all $k \in [0, q - 1]$. A closed walk is a walk that starts and terminates at the same vertex. A closed walk is called a cycle if all the vertices in the walk are distinct, except for the start-vertex and the end-vertex.

The dependency graph of a BCN is defined by $G = (V, E)$, where $V = [X_1, \ldots, X_n, U_1, \ldots, U_p]$, i.e., every vertex corresponds to either an SV or an input of the BCN. The edge $(X_i \rightarrow X_j) \in E$ iff $X_i(k) \mid U_j(k)$. This is an argument of $f_j$ (the update function of $X_i(k+1)$). Thus, the dependency graph encodes the actual variable dependencies in the update functions.

We denote the subset of nodes corresponding to SVs by $V_s := [X_1, \ldots, X_n] \subseteq V$, the subset of edges involving only SVs by $E_s \subseteq E$, and let $G_s := (V_s, E_s)$ be the resulting “reduced” dependency graph. Referring to the outputs of the BCN, we say that a node in the dependency graph that represents a [non] directly observable SV is a [non] directly observable node.

We follow the notion of observability used in [10] and [36].

**Definition 1.** We say that the BCN (1) is observable on $[0, N]$ if for every control sequence $U := \{U(0), \ldots, U(N-1)\}$ and for any two different initial conditions $X(0)$ and $\bar{X}(0)$ the corresponding solutions of the BCN yield different output sequences $\{Y(0), \ldots, Y(N)\}$ and $\{\bar{Y}(0), \ldots, \bar{Y}(N)\}$. The BCN is called observable if it is observable for some value $N \geq 0$.

This means that regardless of the control it is always possible to uniquely determine the initial condition from the output sequence on $[0, N]$.

III. MAIN RESULTS

From here on, we consider BCNs in the form:

$$X_i(k+1) = f_i(X(k), U(k)), \quad i \in [1, n],$$

$$Y_j(k) = X_j(k), \quad j \in [1, m],$$

(2)

that is, every output $Y_j$ is the value of an SV. We assume without loss of generality that the $m$ outputs correspond to the first $m$ SVs. Thus, nodes $X_1, \ldots, X_m \in [X_{m+1}, \ldots, X_n]$ in the dependency graph are [non] directly observable.
Note that a BCN with a general output, namely,
\[ X_i(k+1) = f_i(X(k), U(k)), \quad i \in [1, n], \]
\[ Y_j(k) = g_j(X_1(k), \ldots, X_n(k)), \quad j \in [1, m], \quad (3) \]
can be reduced to the form (2) via a simple augmentation process described in [33].

A. Sufficient Condition for Observability

We now generalize the results derived in [33] for CBNs to general BCNs. We begin by presenting two definitions.

**Definition 2:** We say that a BCN has Property \( P_1 \) if for every non-directly observable node \( X_i \) there exists some other node \( X_j \) such that \( N_{in}(X_i) = \{X_j\} \).

In this case, \( X_i(k+1) = f_i(X_i(k)) \), and thus either \( X_i(k+1) = X_j(k) \) or \( X_i(k+1) = \tilde{X}_j(k) \). This means that the information on the state of \( X_i \) “propagates” to \( X_j \).

**Definition 3:** We say that a BCN has Property \( P_2 \) if every cycle \( C \) in its dependency graph that is composed solely of non-directly observable nodes satisfies the following property: \( C \) includes a node \( X_i \) which is the only element in the in-neighbors set of some other node \( X_j \), i.e., \( N_{in}(X_i) = \{X_j\} \), and \( X_j \) is not part of the cycle \( C \).

This means that \( X_i(k+1) = f_i(X_i(k)) \), so the information on the state of every node in the cycle propagates to \( X_i \) and then to \( X_j \), where \( X_j \) is not part of the cycle. We now provide a sufficient condition for observability.

**Theorem 1:** A BCN that satisfies \( P_1 \) and \( P_2 \) is observable.

To prove this, we first introduce another definition and several auxiliary results.

**Definition 4:** An observed path in the dependency graph is a non-empty ordered set of nodes such that: (1) the last element in the set is a directly observable node; and (2) if the set contains \( p > 1 \) elements, then for any \( i < p \) the \( i \)-th element is a non-directly observable node, and is the only element in the in-neighbors set of node \( i + 1 \). Observed paths with non-overlapping nodes are called *disjoint observed paths* (DOPs).

Roughly speaking, an observed path corresponds to a “shift register” whose last cell is directly observable.

**Proposition 1:** Consider a BCN that satisfies properties \( P_1 \) and \( P_2 \). Then:

1) \( G_s \) can be decomposed into DOPs, such that every vertex in \( G_s \) belongs to a single observed path (i.e., the union of the DOPs is a vertex cover of \( G_s \)).

2) For every vertex \( v \in (V \setminus V_s) \), \( N_{out}(v) \) contains only vertices that are located at the beginning of observed paths.

**Proof:** We give a constructive proof. Algorithm 1 below accepts a graph \( G \) that satisfies properties \( P_1 \) and \( P_2 \) and terminates after each vertex in \( G_s \) belongs to exactly one observed path. Comments in the algorithm are enclosed within (*...*). *-node and *-path are internal variables of the algorithm used to sequentially construct observed paths.

We now prove the correctness of Algorithm 1. To simplify the notation, we say that \( v_p \) points to \( v_q \) if \( p \neq q \) and \( N_{in}(v_q) = \{v_p\} \), and denote this by \( v_p \rightarrow v_q \). The special arrow indicates that the dependency graph includes an edge from \( v_p \) to \( v_q \) and that there are no other edges pointing to \( v_q \).

If all the nodes of \( G_s \) are directly observable (i.e., if \( m = n \)) the algorithm will assign every node to a different observed path and this is correct. Thus, we may assume that \( m < n \). Pick a non directly observable node \( X_j \). Then \( m < j \leq n \). Our first goal is to prove the following result.

**Claim 1:** The algorithm outputs an observed path that contains \( X_j \).

**Proof:** By Property \( P_1 \), there exists \( k \neq j \) such that \( X_j \rightarrow X_k \). We consider two cases.

**Case 1:** If \( k \leq m \) then \( X_k \) is directly observable and the algorithm will add \( X_j \) to an observed path as it “traces back” from \( X_k \) unless \( X_j \) has already been included in some other observed path found by the algorithm. Thus, in this case Claim 1 holds.

**Case 2:** Suppose that \( k > m \), i.e., \( X_k \) is non directly observable. By Property \( P_1 \), there exists \( h \neq k \) such that \( X_k \rightarrow X_h \), so \( X_j \rightarrow X_k \rightarrow X_h \). If \( h \leq m \) then we conclude as in Case 1 that the algorithm outputs an observed path that contains \( X_j \). Thus, we only need to consider the case where as we proceed from \( X_j \) using Property \( P_1 \) we never “find” a directly observable node. Then there exists a set of non directly observable nodes \( X_{k_1}, \ldots, X_{k_l} \), with \( k_l = j \), such that \( X_{k_1} \rightarrow X_{k_2} \rightarrow \cdots \rightarrow X_{k_l} \rightarrow X_i \). This means that \( X_j \) is part of a cycle \( C \) of non directly observable nodes. By Property \( P_2 \), \( C \) includes a node \( X_k \) such that \( X_k \rightarrow X_{si} \), where \( X_{si} \notin C \). If \( X_{si} \) is directly observable then we conclude that the algorithm will output an observed path that includes \( X_j \). If \( X_{si} \) is not directly observable then by Property \( P_1 \), there exists \( s_2 \neq s_1 \) such that \( X_{s_1} \rightarrow X_{s_2} \). Furthermore, since every node in \( C \) has in degree one, \( X_{s_2} \notin C \). Proceeding this way, we conclude that there exist \( s_1, \ldots, s_p \) such that \( X_{s_1} \rightarrow X_{s_2} \rightarrow \cdots \rightarrow X_{s_p} \), with \( X_{s_p} \) a directly observable node. This means that the algorithm will output \( X_j \) in an observed path as it traces back from \( X_{s_p} \), unless it already included \( X_j \) in another observed path. This completes the proof of Claim 1.

Summarizing, we showed that *every* non directly observable node \( X_j \) is contained in an observed path produced by the algorithm. The fact that every \( X_j \) will be in a single
observed path, and that the observed paths will be distinct is clear from the description of the algorithm. From the definition of an observed path, it is clear that only a vertex which is located at the beginning of an observed path may contain edges from vertices in \( V \setminus V_s \). This completes the proof of Proposition 1.

We can now prove Theorem 1.

**Proof of Theorem 1:** Consider the following three statements:

(a) The dependency graph \( G \) has Properties \( P_1 \) and \( P_2 \).

(b) There exists a decomposition of the dependency graph \( G_s \) into \( m \geq 1 \) DOPs \( O^1, \ldots, O^m \), such that every vertex in \( G_s \) belongs to a single observed path; and for every vertex \( v \in (V \setminus V_s) \), \( N_{out}(v) \) contains only vertices that are located at the beginning of observed paths.

(c) The BCN is observable.

The correctness of Algorithm 1 implies that \( (a) \rightarrow (b) \). We now show that \( (b) \rightarrow (c) \).

The initial condition of every SV in \( G_s \) is located at the beginning of an observed path may contain edges from vertices in \( V \setminus V_s \). This completes the proof of Theorem 1.

**Example 1:** Consider the single-input and two-output BCN:

\[
\begin{align*}
X_1(k+1) &= X_4(k), \\
X_2(k+1) &= X_5(k), \\
X_3(k+1) &= (X_3(k) \lor \bar{X}_5(k)) \land \bar{X}_3(k), \\
X_4(k+1) &= X_5(k), \\
X_5(k+1) &= (U_1(k) \lor \bar{X}_1(k)) \land X_3(k), \\
Y_1(k) &= X_1(k), \\
Y_2(k) &= X_2(k).
\end{align*}
\]

The dependency graph \( G \) of this BCN is depicted in Fig. 1. It is straightforward to verify that it satisfies Properties \( P_1, P_2 \). Theorem 1 implies that this BCN is observable, and that \( G_s \) is decomposable into a set of DOPs. Applying Algorithm 1 yields a decomposition into two observed paths: \( O^1 = (X_5, X_4, X_1) \), \( O^2 = (X_3, X_2) \), where \( X_5 \leftrightarrow X_4 \leftrightarrow X_1 \), \( X_3 \leftrightarrow X_2 \).

**Complexity Analysis of the Construction of a Disjoint-Path Observer:** The complexity of generating the dependency graph \( G \) is linear in the description of the graph, which is \( O(|V| + |E|) \). The resulting graph satisfies \( |V| = n + p \), \( |E| \leq n^2 + pn \), with \( n \) being the number of SVs and \( p \) the number of control inputs. By using a chained-list data structure for representing the dependency graph (i.e., each node points to a list of its in-neighbors), Algorithm 1 can be implemented in \( O(n) \) time. Indeed Algorithm 1 systematically passes through each vertex of \( V_s \) once (with \( |V_s| = n \)) and performs \( O(1) \) operations on each such vertex. After the decomposition into DOPs, determining the initial condition of the BCN is attained in complexity which is at most linear in the length of the description of the BCN. From this stage, determination of the state at following time steps is obtained by direct calculation of the dynamics, which is again linear in length of the description of the BCN for every time step.

Summarizing, the complexity of the construction is linear in the length of the description of the BCN, namely \( O(|V| + |E|) \). Thus, it satisfies the bound \( O(n^2 + pn) \).

**C. Output Selection and an Upper Bound for Minimal Observability**

In some cases, it is possible to add sensors to measure more SVs. This motivates the following problem.

**Problem 1:** Given a BCN with \( n \) SVs determine a minimal set of indices \( I \) such that making each \( X_i(k), i \in I \), directly measurable yields an observable BCN.

As mentioned in Section I, this minimal observability problem in BCNs is NP-hard. We use the conditions in Theorem 1 to provide a sub-optimal yet nontrivial solution to Problem 1. This is described by Algorithm 2 which is...
Algorithm 2 Output Selection for Meeting the Condition of Theorem 1: A High-Level Description

**Input:** A BCN (2) with \( n \) SVs and \( m \geq 0 \) outputs.

**Output:** A set of SVs so that making these SVs directly observable yields a BCN that satisfies conditions \( P_1 \) and \( P_2 \).

Theorem 1: A High-Level Description

Three lists. A list \( L \) solution to Problem 1, but rather it solely gives a solution of another node; a list \( L' \) of all SVs that are not directly observable yields a BCN that satisfies conditions \( P_1 \) and \( P_2 \).

Algorithm 2 is based on an algorithm given in [33] that solves the minimal observability problem in a special case of BCNs. In our case (general BCNs) it does not offer a minimal solution to Problem 1, but rather it solely gives a solution which satisfies the conditions in Theorem 1.

The idea behind the algorithm is simple: it first creates three lists. A list \( L_1 \) of all SVs that are not directly observable and are not the only element in the in-neighbors’ set of another node; a list \( L_2 \) of all SVs that are not directly observable and are the only element in the in-neighbors set of another node; and a list \( L_C \) of cycles composed solely out of nodes in \( L_2 \). For each cycle \( C \in L_C \), it then checks if one of its elements appears as the only element in the in-neighbors set of another node that is not part of \( C \). If so, it removes \( C \) from \( L_C \). Finally, it returns the SVs in \( L_1 \) and one element from each cycle \( C \in L_C \). Making these SVs directly observable, by adding them as outputs, yields an observable BCN.

Since the steps of the algorithm are basically described in [33], yet here it is used for general BCNs, we describe only the functional change that is depicted in the headline (i.e., input-output description).

The complexity analysis of Algorithm 2 is done in [33], yielding a runtime which is linear in the length of the description of the BCN, namely \( O(|V| + |E|) \). We note that the algorithm provides a solution which meets the conditions of Theorem 1, but it is straightforward to modify this so that the algorithm will return the information needed to build various possible solutions which meet the conditions of theorem (this is explained in detail in [33]). If the algorithm returns an output list that is empty then the BCN is observable, so it can also be used to determine if a given BCN satisfies the sufficient condition for observability.

**Theorem 2:** Algorithm 2 provides a solution that satisfies the conditions of Theorem 1.

The proof of Theorem 2 is similar to the proof in [33] (but this time it has a different meaning, as here we consider general BCNs), so we omit it. The correctness of Algorithm 2 implies the following.

**Corollary 2:** An upper bound on the minimal size of the solution to Problem 1 is given by the size of the solution generated by Algorithm 2.

Note that for the particular case of CBNs the upper bound provided by size of the solution generated by Algorithm 2 is tight.

### D. Two Examples

Faure et al. [8], [9] derived a BCN model for the mammalian cell cycle that includes nine SVs representing the activity/inactivity of nine different proteins and a single input corresponding to the activation/inactivation of a regulating protein in the cell. The BCN dynamics is given by:

\[
\begin{align*}
    x_1(t + 1) &= (\bar{u}(t) \land \bar{x}_1(t) \land \bar{x}_3(t) \land \bar{x}_4(t) \land \bar{x}_5(t)) \\
    x_2(t + 1) &= (x_1(t) \land \bar{x}_4(t) \land \bar{x}_6(t) \land (x_5(t) \land \bar{x}_1(t) \land \bar{x}_9(t))) \\
    x_3(t + 1) &= x_2(t) \land \bar{x}_1(t), \\
    x_4(t + 1) &= (x_2(t) \land \bar{x}_1(t) \land \bar{x}_6(t) \land (x_7(t) \land \bar{x}_8(t))) \\
    x_5(t + 1) &= (\bar{u}(t) \land \bar{x}_3(t) \land \bar{x}_4(t) \land \bar{x}_9(t)) \\
    x_6(t + 1) &= x_9(t), \\
    x_7(t + 1) &= (x_4(t) \land \bar{x}_9(t) \land \bar{x}_6(t) \land (x_5(t) \land \bar{x}_9(t))) \\
    x_8(t + 1) &= \bar{x}_6(t) \land (x_7(t) \land \bar{x}_8(t) \land (x_6(t) \land \bar{x}_4(t) \land \bar{x}_9(t))). \\
\end{align*}
\]

The input is considered constant in [8], that is, either \( u(t) \equiv True \) or \( u(t) \equiv False \). Under this assumption, the BCN becomes two BNs. The simulations in [8] show that when \( u(t) \equiv True \) the corresponding BN admits a globally attracting periodic trajectory composed of 7 states. The sequence of state transitions along this trajectory qualitatively matches cell cycle progression. For \( u(t) \equiv False \) the BN admits a single state that is globally attracting. This state corresponds to the G0 phase (cell quiescence).

Reference [23] used the STP representation combined with a trial and error approach to conclude that in the particular case where \( u(t) \equiv True \) the solution to the minimal observability problem is to make 8 SV directly measurable, namely, \( X_1, \ldots, X_8 \). Note that using the STP is applicable here, as there are only nine SVs yet the transition matrix is already \( 2^9 \times 2^9 \).

Applying Algorithm 2 to the BCN (5) (without assuming a necessarily constant control) generates the lists: \( L_1 = \{X_1, \ldots, X_8\} \), \( L_2 = \{X_9\} \), and \( L_C = \emptyset \), and then returns \( L_1 \). Thus, in this particular case Algorithm 2 provides an optimal solution.

Our second example is the BCN model of T-cell receptor kinetics from [20]. This has 37 SVs and 3 control inputs. It has been shown that the minimal observability problem for this BCN has a unique solution comprising of 16 directly measurable SVs [36]. Applying Algorithm 2 to this BCN yields a solution with 17 SVs, that includes the 16 SVs in the solution from [36] plus one additional SV called \( Fyn \). Thus, in this case our approach, that only takes into account the network structure, yields a near-optimal solution.

To illustrate the applicability of our results to large-scale networks, we ran Algorithm 2 on randomly generated BCNs. Using a MATLAB implementation on a standard PC (Intel core i5 processor, 4GB RAM memory) the typical runtime for a BCN with \( n = 10^4 \) SVs is a few seconds. We emphasize again that other algorithms, that require exponential run-time,
are not applicable for BCNs with more than a few dozens of SVs.

IV. Conclusion

The problems of analyzing observability, minimal observability, and observer design for general BCNs are NP-hard and existing solutions require an exponential runtime. We described sub-optimal algorithms that have polynomial complexity. This makes them feasible for large-scale BCNs that are typical in systems biology and other scientific fields. To the best of the authors knowledge, these are the first algorithms to address these problems that are applicable to large-scale BCNs. These algorithms are non-trivial in the sense that for a special class of BCNs, namely CBNs, they provide optimal solutions.

An interesting direction for further research is to improve these algorithms so that they provide an optimal solution for a larger sub-class of BCNs.

Acknowledgment

The authors are grateful to the AE and to the anonymous reviewers for very valuable comments.

References

[1] R. Albert and H. G. Othmer, “The topology of the regulatory interactions predicts the expression pattern of the segment polarity genes in Drosophila melanogaster,” J. Theor. Biol., vol. 223, no. 1, pp. 1–18, 2003.
[2] D. Biswas and B. Genest, “Minimal observability for transactional hierarchical services,” in Proc. 20th Int. Conf. Softw. Eng. Knowl. Eng. (SEKE), San Francisco, CA, USA, 2008, pp. 531–536.
[3] X. Chen, Z. Gao, and T. Basar. (2017). Asymptotic Behavior of Conjunctive Boolean Networks Over Weakly Connected Digraphs. [Online]. Available: https://arxiv.org/abs/1708.01975
[4] D. Cheng, “Semi-tensor product of matrices and its application to Morgen’s problem,” Sci. China Inf. Sci., vol. 44, no. 3, pp. 195–212, 2001.
[5] D. Cheng, “Disturbance decoupling of Boolean control networks,” IEEE Trans. Autom. Control, vol. 56, no. 1, pp. 1–20, Jan. 2011.
[6] D. Cheng, C. Li, and F. He, “Observability of Boolean networks via set controllability approach,” Syst. Control Lett., vol. 115, pp. 22–25, May 2018.
[7] D. Cheng and H. Qi, “Controllability and observability of Boolean control networks,” Automatica, vol. 45, no. 7, pp. 1659–1667, 2009.
[8] A. Faure, A. Naldi, C. Chauviery, and D. Thieffry, “Dynamical analysis of a generic Boolean model for the control of the mammalian cell cycle,” Bioinformatics, vol. 22, no. 14, pp. e124–e131, 2006.
[9] A. Faure and D. Thieffry, “Logical modelling of cell cycle control in eukaryotes: A comparative study,” Mol. Biol.Syst., vol. 5, no. 12, pp. 1569–1581, 2009.
[10] E. Fornasini and M. E. Valcher, “Observability, reconstructability and state observers of Boolean control networks,” IEEE Trans. Autom. Control, vol. 58, no. 6, pp. 1390–1401, Jun. 2013.
[11] E. Fornasini and M. E. Valcher, “Optimal control of Boolean control networks,” IEEE Trans. Autom. Control, vol. 59, no. 5, pp. 1258–1270, May 2014.
[12] Z. Gao, X. Chen, and T. Basar, “Controllability of conjunctive Boolean networks with application to gene regulation,” IEEE Trans. Control Netw. Syst., vol. 5, no. 2, pp. 770–781, Jun. 2018.
[13] Z. Gao, X. Chen, and T. Basar, “Stability structures of conjunctive Boolean networks,” Automatica, vol. 89, pp. 8–20, Mar. 2018.
[14] D. G. Green, T. G. Leishman, and S. Sadedin, “The emergence of social consensus in Boolean networks,” in Proc. IEEE Symp. Artif. Life (IEEE-ALife), Honolulu, HI, USA, 2007, pp. 402–408.
[15] X. Hulhoven, A. V. Wouwer, and P. Bogaerts, “Hybrid extended Luenberger-asymptotic observer for biprocess state estimation,” Chem. Eng. Sci., vol. 61, no. 21, pp. 7151–7160, 2006.
[16] I. Iruzun-Aran, J. M. Pastor, I. F. Trocóniz, and J. D. Gómez-Mantilla, “Advanced Boolean modeling of biological networks applied to systems pharmacology,” Bioinformatics, vol. 33, no. 7, pp. 1040–1048, 2017.
[17] R. M. Jungers and V. D. Blondel, “Observable graphs,” Discr. Appl. Math., vol. 159, no. 10, pp. 981–989, 2011.
[18] A. Kasyanov, L. Kirkland, and M. T. Mihalea, “A spatial SIRS Boolean network model for the spread of H5N1 avian influenza virus among poultry farms,” in Proc. 5th Int. Workshop Comput. Syst. Biol. (WCSB), 2008, pp. 73–76.
[19] S. A. Kauffman, “Metabolic stability and epigenesis in randomly constructed genetic nets,” J. Theor. Biol., vol. 22, no. 3, pp. 437–467, 1969.
[20] Z. Zhang, T. Leifeld, and P. Zhang, “Observer design for Boolean control networks,” Proc. Design Autom. Test Europe Conf. (DATE), 2018, pp. 1–6.
[21] S. Shahrampour and V. M. Preciado, “Topology identification of directed dynamical networks via power spectral analysis,” IEEE Trans. Autom. Control, vol. 60, no. 8, pp. 2260–2265, Aug. 2015.
[22] E. Sontag, “Distributed Control Theory: Deterministic Finite Dimensional Systems,” 2nd ed. New York, NY, USA: Springer, 1998.
[23] R.-S. Wang, A. Saadatpour, and R. Albert, “Boolean modeling in systems biology: An overview of methodology and applications,” Phys. Biol., vol. 9, no. 5, 2012, Art. no. 055001.
[24] E. Weiss and M. Margaliot, “A polynomial-time algorithm for solving the minimal observability problem in conjunctive Boolean networks,” IEEE Trans. Autom. Control, to be published. [Online]. Available: arXiv preprint arXiv:1706.04072
[25] E. Weiss, M. Margaliot, and G. Even, “Minimal controllability of conjunctive Boolean networks is NP-complete,” Automatica, vol. 92, pp. 56–62, Jun. 2018.
[26] K. Zhang and L. Zhang, “Observer design for Boolean control networks,” IEEE Trans. Autom. Control, vol. 61, no. 9, pp. 2733–2738, Sep. 2016.
[27] K. Zhang, “Observability and reconstructibility of large-scale Boolean control networks via network aggregations,” arXiv preprint arXiv:1704.03231, 2017.
[28] Z. Zhang, T. Leifeld, and P. Zhang, “Finite horizon tracking control of Boolean control networks,” IEEE Trans. Autom. Control, vol. 63, no. 6, pp. 1798–1805, Jun. 2018.
[29] Z. Zhang, T. Leifeld, and P. Zhang, “Observer design for Boolean control networks,” in Proc. 55th IEEE Conf. Decis. Control (CDC), 2016, pp. 6272–6277.