Finite-horizon optimal control of Boolean control networks

Ettore Fornasini and Maria Elena Valcher

Abstract

In this paper we address the finite-horizon optimal control problem for Boolean control networks (BCNs). By resorting to the algebraic approach recently introduced by D. Cheng and co-authors [1], [3], [4], [5], we first pose the problem of finding the input sequences that minimize a given quadratic cost function. Then, by resorting to the semi-tensor product, we rewrite the cost function as a linear one. The problem solution is obtained by means of a recursive algorithm that represents the analogue for BCNs of the difference Riccati equation for linear systems. A number of apparently more general optimal control problems for BCNs can be easily reframed into the present set-up. In particular, the cost function can be adjusted so as to include penalties on the switchings, provided that we redefine the BCN state variable.

I. Introduction

Boolean networks (BNs) have recently witnessed a renewed interest. BNs have proved to be an effective tool for representing a number of phenomena whose describing variables display only two operation levels. This is the case of genetic regulation networks [10], that can be successfully modeled as BNs, due to the fact that each gene can be naturally described as a binary device that displays only two states: "transcribed" or "quiescent", namely "on" or "off". BNs have also been used to describe the interactions among agents, and hence to investigate consensus problems [9].

On the other hand, when the network behavior depends also on some (Boolean) control inputs, the concept of BN, whose dynamics is uniquely determined once the initial conditions

Ettore Fornasini and Maria Elena Valcher are with the Dipartimento di Ingegneria dell'Informazione, Università di Padova, via Gradenigo 6/B, 35131 Padova, Italy, {fornasini,meme}@dei.unipd.it.

are assigned, naturally extends to that of Boolean control network (BCN). A BCN can be seen as a family of BNs, each of them associated with a specific value of the input variables, and in that sense it reminds of a switching system.

In the last decade, an algebraic framework has been developed that casts both BNs and BCNs into the framework of linear state-space models (operating on canonical vectors) [1], [3], [4], [5]. Within this setting, several control problems, like stability, stabilizability [2], [7], [8], controllability [13] and observability [6], have been investigated.

The optimal control of BCNs has been recently addressed in a few contributions. Specifically, in [15] (see also Chapter 15 in [5]) the problem of finding the input sequence that maximizes, on the infinite-horizon, an average payoff that weights both the state and the input at every time $t \in \mathbb{Z}_+$, is investigated. The optimal solution is expressed as a constant feedback law, driving the BCN on a periodic state trajectory with maximum payoff. On the other hand, [11] and [12] consider the optimum control problem on a finite horizon, but restrict the analysis to the case when the payoff function only depends on the state of the BCN at the end of the control interval. The optimum solution is obtained by resorting to the maximum principle, and has the structure of a time varying state feedback law.

By following this stream of research, in this paper we address the finite-horizon optimal control problem for BCNs, by assuming a cost function that depends on both the state and the input values at every time instant. We first consider a quadratic cost function, and, by resorting to the semi-tensor product, we rewrite it as a linear one. The problem solution is obtained by means of a recursive algorithm that represents the analogue for BCNs of the difference Riccati equation for linear systems. A number of apparently more general optimal control problems for BCNs are shown to be easily reframed into the present set-up. In particular, the cost function can be adjusted so as to include penalties on the switchings, provided that the BCN state variable is suitable redefined.

Notation. $\mathbb{Z}_+$ denotes the set of nonnegative integers. Given two integers $k, n \in \mathbb{Z}_+$, with $k \leq n$, by the symbol $[k, n]$ we denote the set of integers $\{k, k+1, \ldots, n\}$. We consider Boolean vectors and matrices, taking values in $\mathcal{B} := \{0, 1\}$, with the usual operations (sum $+$, product $\cdot$ and negation $\neg$).

δ_k^i will denote the ith canonical vector of size k, $\mathcal{L}_k$ the set of all k-dimensional canonical

vectors, and $\mathcal{L}_{k\times n} \subset \mathcal{B}^{k\times n}$ the set of all $k \times n$ matrices whose columns are canonical vectors of size k. Any matrix $L \in \mathcal{L}_{k\times n}$ can be represented as a row vector whose entries are canonical vectors in $\mathcal{L}_k$, namely $L = [\,\delta_k^{i_1} \quad \delta_k^{i_2} \quad \dots \quad \delta_k^{i_n}\,]$, for suitable indices $i_1, i_2, \dots, i_n \in [1, k]$. The k-dimensional vector with all entries equal to 1, is denoted by $\mathbf{1}_k$.

The (ℓ, j)th entry of a matrix L is denoted by $[L]_{\ell j}$, while the ℓth entry of a vector $\mathbf{v}$ is $[\mathbf{v}]_\ell$. The ith column of a matrix L is $\mathrm{col}_i(L)$.

There is a bijective correspondence between Boolean variables $X \in \mathcal{B}$ and vectors $\mathbf{x} \in \mathcal{L}_2$, defined by the relationship

$$\mathbf{x} = \begin{bmatrix} X \\ \neg X \end{bmatrix}.$$

We introduce the *(left) semi-tensor product* $\ltimes$ between matrices (and hence, in particular, vectors) as follows [5], [13], [14]: given $L_1 \in \mathbb{R}^{r_1\times c_1}$ and $L_2 \in \mathbb{R}^{r_2\times c_2}$ (in particular, $L_1 \in \mathcal{L}_{r_1\times c_1}$ and $L_2 \in \mathcal{L}_{r_2\times c_2}$), we set

$$L_1 \ltimes L_2 := (L_1 \otimes I_{T/c_1})(L_2 \otimes I_{T/r_2}), \qquad T := \text{l.c.m.}\{c_1, r_2\},$$

where l.c.m. denotes the least common multiple. The semi-tensor product represents an extension of the standard matrix product, by this meaning that if $c_1 = r_2$, then $L_1 \ltimes L_2 = L_1 L_2$. Note that if $\mathbf{x}_1 \in \mathcal{L}_{r_1}$ and $\mathbf{x}_2 \in \mathcal{L}_{r_2}$, then

$$\mathbf{x}_1 \ltimes \mathbf{x}_2 \in \mathcal{L}_{r_1 r_2}.$$

For the various properties of the semi-tensor product we refer to [5]. By resorting to the semi-tensor product, we can extend the previous correspondence to a bijective correspondence between $\mathcal{B}^n$ and $\mathcal{L}_{2^n}$. This is possible in the following way: given $X = [\,X_1 \quad X_2 \quad \dots \quad X_n\,]^\top \in \mathcal{B}^n$, set

$$\mathbf{x} := \begin{bmatrix} X_1 \\ \neg X_1 \end{bmatrix} \ltimes \begin{bmatrix} X_2 \\ \neg X_2 \end{bmatrix} \ltimes \dots \ltimes \begin{bmatrix} X_n \\ \neg X_n \end{bmatrix},$$

or equivalently

$$\mathbf{x} = \begin{bmatrix} X_1 X_2 \dots X_{n-1} X_n \\ X_1 X_2 \dots X_{n-1} \neg X_n \\ X_1 X_2 \dots \neg X_{n-1} X_n \\ \vdots \\ \neg X_1 \neg X_2 \dots \neg X_{n-1} \neg X_n \end{bmatrix}.$$

II. Optimal control of BCNs: problem statement

A *Boolean control network* (BCN) is described by the following equations

$$\begin{aligned} X(t+1) &= f(X(t), U(t)), \\ Y(t) &= h(X(t)), \qquad t \in \mathbb{Z}_+, \end{aligned} \tag{1}$$

where $X(t)$, $U(t)$ and $Y(t)$ denote the n-dimensional state variable, the m-dimensional input and the p-dimensional output at time t, taking values in $\mathcal{B}^n, \mathcal{B}^m$ and $\mathcal{B}^p$, respectively. f, h are (logic) functions, i.e. $f : \mathcal{B}^n \times \mathcal{B}^m \to \mathcal{B}^n$ and $h : \mathcal{B}^n \to \mathcal{B}^p$. By resorting to the semi-tensor product $\ltimes$, state, input and output Boolean variables can be represented as canonical vectors in $\mathcal{L}_N,\ N = 2^n,\ \mathcal{L}_M,\ M = 2^m$, and $\mathcal{L}_P,\ P = 2^p$, respectively, and the BCN (1) satisfies [5] the following *algebraic description*:

$$\begin{aligned} \mathbf{x}(t+1) &= L \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t), \qquad t \in \mathbb{Z}_+, \\ \mathbf{y}(t) &= H\mathbf{x}(t) \end{aligned} \tag{2}$$

where $\mathbf{x}(t) \in \mathcal{L}_N, \mathbf{u}(t) \in \mathcal{L}_M$ and $\mathbf{y}(t) \in \mathcal{L}_P$. $L \in \mathcal{L}_{N \times NM}$ and $H \in \mathcal{L}_{P \times N}$ are matrices whose columns are all canonical vectors of size N and P, respectively. For every choice of the input variable at t, namely for every $\mathbf{u}(t) = \delta_M^j$, $L \ltimes \mathbf{u}(t) =: L_j$ is a matrix in $\mathcal{L}_{N \times N}$. So, we can think of the state equation of the BCN (2) as a Boolean switched system,

$$\mathbf{x}(t+1) = L_{\sigma(t)}\mathbf{x}(t), \qquad t \in \mathbb{Z}_+, \tag{3}$$

where $\sigma(t), t \in \mathbb{Z}_+$, is a switching sequence taking values in $[1, M]$. For every $j \in [1, M]$, we refer to the BN

$$\mathbf{x}(t+1) = L_j\mathbf{x}(t), \qquad t \in \mathbb{Z}_+, \tag{4}$$

as to the jth subsystem of the Boolean switched system (3). Note that the matrix L can be expressed in terms of the matrices L_js as:

$$L = [\, L_1 \quad L_2 \quad \dots \quad L_M \,].$$

By referring to the algebraic description of the BCN (2), we introduce the following **finite-horizon optimal control problem**, which is inspired by the interpretation of the BCN as a Boolean switched system and assumes a quadratic cost function (see Problem 1 in [16]):

Given the BCN (2), with initial state $\mathbf{x}(0) = \mathbf{x}_0 \in \mathcal{L}_N$*, determine an input sequence that minimizes the quadratic cost function:*

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{x}(T)^\top Q_f \mathbf{x}(T) + \sum_{t=0}^{T-1} \mathbf{x}(t)^\top Q_{\mathbf{u}(t)} \mathbf{x}(t), \tag{5}$$

where $Q_f = Q_f^\top \in \mathbb{R}^{N \times N}$ *is a quadratic positive semi-definite matrix, and* $Q_{\mathbf{u}(t)} = Q_{\mathbf{u}(t)}^\top$ *is a quadratic positive semi-definite matrix function that takes a specific value in* $\mathbb{R}^{N \times N}$ *for every value of the input sample* $\mathbf{u}(t)$ *in* $\mathcal{L}_M$*.*

This quadratic cost function weights the BCN state at every time instant: the final state is weighted by a special weight matrix Q_f, while the state at every intermediate instant t is weighted by a quadratic matrix that depends on the input value at the same time t. So, overall, it can be regarded as a "switched quadratic function", that depends on the specific "switched sequence" $\mathbf{u}(t), t \in [0, T-1]$. For the sake of simplicity, we use the compact notation $Q_i := Q_{\delta_M^i}$, $\forall\ i \in [1, M]$.

The first step to take is to show that the switched quadratic cost function can always be equivalently expressed as a switched linear cost function, by resorting to the semi-tensor product. Indeed, since $\mathbf{x}(T)$ is a canonical vector in $\mathcal{L}_N$, we have that

$$\mathbf{x}(T) = \delta_N^j \qquad \Rightarrow \qquad \mathbf{x}(T)^\top Q_f \mathbf{x}(T) = [Q_f]_{jj}.$$

Therefore

$$\mathbf{x}(T)^\top Q_f \mathbf{x}(T) = \mathbf{c}_f^\top \mathbf{x}(t),$$

for $\mathbf{c}_f^\top := [\,[Q_f]_{11} \quad [Q_f]_{22} \quad \dots \quad [Q_f]_{NN}\,]$. Similarly, by making use of the semi-tensor product properties, and of the fact that $\mathbf{x}(t) \in \mathcal{L}_N$ and $\mathbf{u}(t) \in \mathcal{L}_M$, for every $t \in [0, T-1]$, we have that

$$\left.\begin{array}{l} \mathbf{x}(t) = \delta_N^j \\ \mathbf{u}(t) = \delta_M^i \end{array}\right\} \Rightarrow \mathbf{x}(t)^\top Q_{\mathbf{u}(t)} \mathbf{x}(t) = [Q_i]_{jj} = \mathbf{c}^\top \ltimes \delta_M^i \ltimes \delta_N^j,$$

where

$$\begin{aligned} \mathbf{c}^\top &:= [\,[Q_1]_{11} \quad \dots \quad [Q_1]_{NN} \mid \dots \mid [Q_M]_{11} \quad \dots \quad [Q_M]_{NN}\,] \\ &= [\,\mathbf{c}_1^\top \mid \dots \mid \mathbf{c}_M^\top\,]. \end{aligned}$$

This implies that the index (5) can be equivalently rewritten as follows:

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{c}_f^\top \mathbf{x}(T) + \sum_{t=0}^{T-1} \mathbf{c}^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t), \tag{6}$$

where $\mathbf{c}_f \in \mathbb{R}^N$ and $\mathbf{c} \in \mathbb{R}^{NM}$.

Remark 1: The assumptions we initially introduced on the matrices Q_f and $Q_i, i \in [1, M]$, did not play any role in proving that the cost index (5) may be rewritten as in (6). In fact, they could be arbitrary matrices in $\mathbb{R}^{N\times N}$. The positive semidefiniteness of Q_f and $Q_i, i \in [1, M]$, ensures only the nonnegativity of the vectors $\mathbf{c}_f$ and $\mathbf{c}$.

On the other hand, as the input and state vectors take only a finite set of values, it is easily seen that the input sequence that minimizes the cost function (6) (for a given $\mathbf{x}_0$) is the same one that minimizes

$$\begin{aligned}\tilde{J}(\mathbf{x}_0, \mathbf{u}(\cdot)) &= [\mathbf{c}_f + \alpha \mathbf{1}_N]^\top \mathbf{x}(T) \\ &+ \sum_{t=0}^{T-1} [\mathbf{c} + \beta \mathbf{1}_{NM}]^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t),\end{aligned}$$

for any choice of $\alpha, \beta \in \mathbb{R}$. So, it is always possible to assume that the weight vectors $\mathbf{c}_f$ and $\mathbf{c}_i, i \in [1, M]$, are nonnegative, without affecting the optimal control solution.

Therefore, in the rest of the paper, we find the input trajectory that minimizes the index cost (6) for the BCN (2), starting form the initial condition $\mathbf{x}(0) = \mathbf{x}_0$.

III. Optimal control of BCNs: problem solution

We first observe that for every choice of a family of real vectors $\mathbf{m}(t), t \in [0, T]$, and every state trajectory of the BCN $\mathbf{x}(t), t \in [0, T]$, one has

$$\begin{aligned}0 &= \sum_{t=0}^{T-1} [\mathbf{m}(t+1)^\top \mathbf{x}(t+1) - \mathbf{m}(t)^\top \mathbf{x}(t)] \\ &+ \mathbf{m}(0)^\top \mathbf{x}(0) - \mathbf{m}(T)^\top \mathbf{x}(T).\end{aligned}$$

Consequently, the cost function (6) can be equivalently written as

$$\begin{aligned}J(\mathbf{x}_0, \mathbf{u}(\cdot)) &= \mathbf{m}(0)^\top \mathbf{x}(0) + [\mathbf{c}_f - \mathbf{m}(T)]^\top \mathbf{x}(T) \\ &+ \sum_{t=0}^{T-1} \mathbf{c}^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t) \\ &+ \sum_{t=0}^{T-1} [\mathbf{m}(t+1)^\top \mathbf{x}(t+1) - \mathbf{m}(t)^\top \mathbf{x}(t)].\end{aligned}$$

Now, we make use of the state update equation of the BCN (2) and of the fact that, for every choice of $\mathbf{u}(t) \in \mathcal{L}_M$, one has

$$\mathbf{m}(t)^\top \mathbf{x}(t) = [\,\mathbf{m}(t)^\top \quad \mathbf{m}(t)^\top \quad \dots \quad \mathbf{m}(t)^\top\,] \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t).$$

This way we get

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{m}(0)^\top \mathbf{x}(0) + [\mathbf{c}_f - \mathbf{m}(T)]^\top \mathbf{x}(T)$$

$$+ \sum_{t=0}^{T-1} \left(\mathbf{c}^\top + \mathbf{m}(t+1)^\top L - [\,\mathbf{m}(t)^\top \quad \dots \quad \mathbf{m}(t)^\top\,]\right)$$

$$\ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t).$$

Now, since the vectors $m(t), t \in [0, T]$, can be freely chosen without affecting the value of the index, we choose them according to the following **algorithm**:

- [Initialization] Set $\mathbf{m}(T) := \mathbf{c}_f$;
- [Recursion] For $t = T-1, T-2, \dots, 1, 0$, the jth entry of the vector $\mathbf{m}(t)$ is chosen according to the recursive rule:

$$[\mathbf{m}(t)]_j := \min_{i \in [1,M]} \Big([\mathbf{c}_i]_j + [\mathbf{m}(t+1)^\top L_i]_j \Big), \ \forall\ j \in [1, N].$$

We notice that, by the previous algorithm, for every $t \in [0, T-1]$ the vector

$$\begin{aligned} \mathbf{w}(t)^\top &:= [\,\mathbf{w}_1(t)^\top \quad \mathbf{w}_2(t)^\top \quad \dots \quad \mathbf{w}_M(t)^\top\,] \\ &= [\,\mathbf{c}_1^\top \quad | \quad \dots \quad | \quad \mathbf{c}_M^\top\,] \\ &+ \quad \mathbf{m}(t+1)^\top [\,L_1 \quad L_2 \quad \dots \quad L_M\,] \\ &- [\,\mathbf{m}(t)^\top \quad \mathbf{m}(t)^\top \quad \dots \quad \mathbf{m}(t)^\top\,] \end{aligned}$$

is nonnegative and satisfies the following condition: for every $j \in [1, N]$ there exists $i \in [1, M]$ such that $[\mathbf{w}_i(t)]_j = 0$. As a result, the index

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{m}(0)^\top \mathbf{x}(0)$$

$$+ \sum_{t=0}^{T-1} [\,\mathbf{w}_1(t)^\top \quad \mathbf{w}_2(t)^\top \quad \dots \quad \mathbf{w}_M(t)^\top\,] \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t)$$

is minimized by the input sequence $\mathbf{u}(t), t \in [0, T-1]$ that is obtained according to this rule:

$$\mathbf{x}(t) = \delta_N^j \qquad \Rightarrow \qquad \mathbf{u}(t) = \delta_M^{i^*(j,t)},$$

where[1]

$$i^*(j,t) = \arg \min_{i \in [1,M]} \Big([\mathbf{c}_i]_j + [\mathbf{m}(t+1)^\top L_i]_j \Big).$$

In this way, indeed,

$$[\, \mathbf{w}_1^\top(t) \quad \dots \quad \mathbf{w}_M^\top(t) \,] \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t) = 0, \ \forall \ t \in [0, T-1],$$

and by the nonnegativity of the vector $\mathbf{w}(t)$, this is the minimum possible value that this term can take.

Two straightforward consequences of the previous analysis are:

- $J^*(\mathbf{x}_0) = \min_{\mathbf{u}(\cdot)} J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{m}(0)^\top \mathbf{x}(0)$, where $\mathbf{m}(0)$ is obtained according to the previous algorithm.
- The optimal control input can be implemented by means of a time-varying feedback law. Actually, the optimal input can be expressed as

  $$\mathbf{u}(t) = K(t)\mathbf{x}(t),$$

  where the (not necessarily unique) feedback matrix is expressed as

  $$K(t) = [\, \delta_M^{i^*(1,t)} \quad \delta_M^{i^*(2,t)} \quad \dots \quad \delta_M^{i^*(N,t)} \,].$$

Remark 2: The previous algorithm can be viewed as the equivalent for BCNs of the difference Riccati equation for standard discrete-time linear systems with a quadratic cost function. The algorithm, however, is based on a linear functional instead of a quadratic one, due to the fact that we have replaced the quadratic cost function (5) with an equivalent linear one.

Example 1: Consider the BCN (2) and suppose that $N = 8$, $M = 2$ and

$$\begin{aligned} L_1 &:= L \ltimes \delta_2^1 = [\, \delta_8^4 \quad \delta_8^5 \quad \delta_8^4 \quad \delta_8^5 \quad \delta_8^6 \quad \delta_8^7 \quad \delta_8^8 \quad \delta_8^7 \,], \\ L_2 &:= L \ltimes \delta_2^2 = [\, \delta_8^2 \quad \delta_8^4 \quad \delta_8^1 \quad \delta_8^7 \quad \delta_8^6 \quad \delta_8^5 \quad \delta_8^6 \quad \delta_8^6 \,]. \end{aligned}$$

We consider the problem of minimizing the cost function (6) for $T = 4$, by assuming

$$\begin{aligned} \mathbf{c}_f^\top &= [\, 1 \quad 1 \quad 1 \quad 2 \quad 1 \quad 10 \quad 0 \quad 0 \,], \\ \mathbf{c}^\top &= [\, \mathbf{1}_8^\top \quad \mathbf{0}_8^\top \,], \end{aligned}$$

[1]Note that the index that minimizes the function is not necessarily unique: so there is not necessarily a unique optimal input.

and initial condition $\mathbf{x}(0) = \delta_8^1$.

It is worth noticing that the input $\mathbf{u}(t) = \delta_2^2$ has zero cost. So, one would be tempted to just assume $\mathbf{u}(t) = \delta_2^2$ for every $t \in [0, 3]$. This way, however, $\mathbf{x}(4)$ would be equal to δ_8^6, which is the "most expensive" final state. So, we proceed according to the algorithm:

- $\mathbf{m}(4) = \mathbf{c}_f^\top = [\,1 \quad 1 \quad 1 \quad 2 \quad 1 \quad 10 \quad 0 \quad 0\,]$;
- $\mathbf{m}(3) = [\,1 \quad 2 \quad 1 \quad 0 \quad 10 \quad 1 \quad 1 \quad 1\,]$ and
  $K(3) = [\,\delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^1 \quad \delta_2^1\,]$;
- $\mathbf{m}(2) = [\,1 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1\,]$ and
  $K(2) = [\,\delta_2^1 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2\,]$;
- $\mathbf{m}(1) = [\,0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1\,]$ and
  $K(1) = [\,\delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2\,]$;
- $\mathbf{m}(0) = [\,0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1\,]$ and
  $K(0) = [\,\delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2 \quad \delta_2^2\,]$.

As a consequence,

$$J^* = \min_{\mathbf{u}(\cdot)} J(\delta_8^1, \mathbf{u}(\cdot)) = \mathbf{m}(0)^\top \delta_8^1 = 1.$$

An optimal input sequence is

$$\mathbf{u}^*(0) = \mathbf{u}^*(1) = \mathbf{u}^*(2) = \delta_2^2, \qquad \mathbf{u}^*(3) = \delta_2^1,$$

and it corresponds to the state-trajectory

$$\mathbf{x}^*(0) = \delta_1^8, \qquad \mathbf{x}^*(1) = \delta_2^8, \quad \mathbf{x}^*(2) = \delta_8^4,$$
$$\mathbf{x}^*(3) = \delta_8^7, \qquad \mathbf{x}^*(4) = \delta_8^8.$$

IV. Extensions and special cases of the finite-horizon optimal control problem

A. General quadratic cost function

The choice of the quadratic switched cost function (5) (and hence the associated linear cost function (6)) may seem a restrictive one, however it can be easily proved that the quadratic cost

function commonly used for linear state-space models can be easily reduced to the form (6). Indeed, if we assume

$$\begin{aligned} J(\mathbf{x}_0, \mathbf{u}(\cdot)) &= \mathbf{x}(T)^\top Q_f \mathbf{x}(T) \\ &+ \sum_{t=0}^{T-1} \begin{bmatrix} \mathbf{x}(t)^\top & \mathbf{u}(t)^\top \end{bmatrix} \begin{bmatrix} Q & S \\ S^\top & R \end{bmatrix} \begin{bmatrix} \mathbf{x}(t) \\ \mathbf{u}(t) \end{bmatrix}, \end{aligned} \tag{7}$$

where Q_f, Q and R are symmetric matrices, the index can be first equivalently expressed as

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{c}_f \mathbf{x}(T) + \sum_{t=0}^{T-1} \mathbf{q}^\top \mathbf{x}(t) + \mathbf{r}^\top \mathbf{u}(t) + 2\mathbf{u}^\top S^\top \mathbf{x}(t),$$

where

$$\begin{aligned} \mathbf{c}_f^\top &:= \begin{bmatrix} [Q_f]_{11} & [Q_f]_{22} & \dots & [Q_f]_{NN} \end{bmatrix}, \\ \mathbf{q}^\top &:= \begin{bmatrix} [Q]_{11} & [Q]_{22} & \dots & [Q]_{NN} \end{bmatrix}, \\ \mathbf{r}^\top &:= \begin{bmatrix} [R]_{11} & [R]_{22} & \dots & [R]_{MM} \end{bmatrix}. \end{aligned}$$

On the other hand, upon noticing that

$$\begin{aligned} \mathbf{q}^\top \mathbf{x}(t) &= \begin{bmatrix} \mathbf{q}^\top & \mathbf{q}^\top & \dots & \mathbf{q}^\top \end{bmatrix} \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t), \\ \mathbf{r}^\top \mathbf{u}(t) &= \begin{bmatrix} [\mathbf{r}]_1 \mathbf{1}_N & [\mathbf{r}]_2 \mathbf{1}_N & \dots & [\mathbf{r}]_M \mathbf{1}_N \end{bmatrix} \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t), \\ \mathbf{u}^\top S^\top \mathbf{x}(t) &= \begin{bmatrix} \mathrm{col}_1(S)^\top & \mathrm{col}_2(S)^\top & \dots & \mathrm{col}_M(S)^\top \end{bmatrix} \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t), \end{aligned}$$

we finally rewrite the cost function in the form (6) for

$$\begin{aligned} \mathbf{c}^\top &:= \begin{bmatrix} \mathbf{q}^\top & \mathbf{q}^\top & \dots & \mathbf{q}^\top \end{bmatrix} \\ &+ \begin{bmatrix} [\mathbf{r}]_1 \mathbf{1}_N & [\mathbf{r}]_2 \mathbf{1}_N & \dots & [\mathbf{r}]_M \mathbf{1}_N \end{bmatrix} \\ &+ 2 \begin{bmatrix} \mathrm{col}_1(S)^\top & \mathrm{col}_2(S)^\top & \dots & \mathrm{col}_M(S)^\top \end{bmatrix}. \end{aligned}$$

B. Cost function that weights the outputs instead of the states

The quadratic cost function

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{y}(T)^\top \hat{Q}_f \mathbf{y}(T) + \sum_{t=0}^{T-1} \mathbf{y}(t)^\top \hat{Q}_{\mathbf{u}(t)} \mathbf{y}(t), \tag{8}$$

can be easily brought to the form

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{c}_f^\top \mathbf{x}(T) + \sum_{t=0}^{T-1} \mathbf{c}^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t),$$

where

$$\begin{aligned}\mathbf{c}_f^\top &:= [\,[\hat{Q}_f]_{11} \quad [\hat{Q}_f]_{22} \quad \dots \quad [\hat{Q}_f]_{PP}\,]\,H,\\ \mathbf{c}^\top &:= [\,\hat{\mathbf{c}}_1^\top H \;\mid\; \dots \;\mid\; \hat{\mathbf{c}}_M^\top H\,],\end{aligned}$$

and

$$\begin{aligned}[\,\hat{\mathbf{c}}_1^\top \;\mid\; \dots \;\mid\; \hat{\mathbf{c}}_M^\top\,] \;=\; &[\,[\hat{Q}_1]_{11} \quad \dots \quad [\hat{Q}_1]_{PP}\,\mid\, \dots\\ &\mid [\hat{Q}_M]_{11} \quad \dots \quad [\hat{Q}_M]_{PP}\,].\end{aligned}$$

C. Time-varying weight matrices

The previous analysis immediately extends to the case when the cost function is time-dependent instead of switch-dependent, namely the case (see Problem 2 [16]) when

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{x}(T)^\top Q_f \mathbf{x}(T) + \sum_{t=0}^{T-1} \mathbf{x}(t)^\top Q(t)\mathbf{x}(t), \tag{9}$$

where $Q_f = Q_f^\top \in \mathbb{R}^{N\times N}$ is a positive semi-definite matrix, and $Q(t) = Q(t)^\top$ is a positive semi-definite matrix in $\mathbb{R}^{N\times N}$ for every value of $t \in [0, T-1]$. Indeed, when so, the cost function can be equivalently rewritten as

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{c}_f^\top \mathbf{x}(T) + \sum_{t=0}^{T-1} [\,\mathbf{c}^\top(t) \;\; \dots \; \mathbf{c}^\top(t)\,] \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t),$$

for suitable $\mathbf{c}_f \in \mathbb{R}^N$ and $\mathbf{c}(t) \in \mathbb{R}^N$. So, the whole analysis is a minor modification of the previous one.

D. Cost function that weights only the final state

In a couple of recent papers [12], [11] the finite-horizon optimal control problem for BCNs has been addressed by resorting to Pontryagin maximum principle, and by assuming as cost function a linear function of the final state:

$$J(\mathbf{x}_0, \mathbf{u}(\cdot)) = \mathbf{r}^\top \mathbf{x}(T), \qquad \mathbf{r} \in \mathbb{R}^N. \tag{10}$$

The problem of determining the state trajectory starting from a given $\mathbf{x}(0) = \mathbf{x}_0$ that minimizes the cost function (10) could be cast in our analysis, by simply assuming $\mathbf{c} = 0$. The algorithm becomes the following one:

- [Initialization] Set $\mathbf{m}(T) := \mathbf{c}_f$;
- [Recursion] For $t = T-1, T-2, \ldots, 1, 0$, the jth entry of the vector $\mathbf{m}(t)$ is chosen according to the recursive rule:

$$[\mathbf{m}(t)]_j := \min_{i\in[1,M]} [\mathbf{m}(t+1)^\top L_i]_j, \ \forall\ j \in [1,N].$$

The minimum cost is again $J^*(\mathbf{x}_0) = \mathbf{m}(0)^\top \mathbf{x}_0$, and the corresponding optimal input could be expressed as a time-varying feedback from the (optimal) state trajectory. If we reverse the perspective and instead of minimizing the cost function we aim at maximizing it, as investigated in [12], [11], the algorithm requires a minor modification. Actually, at each recursion step the goal is to select the entries of the vector $\mathbf{m}(t)$ in such a way that for every $j \in [1,N]$ there exists $i \in [1,M]$ such that $[\mathbf{m}(t+1)^\top L_i]_j - [\mathbf{m}(t)]_j = 0$ while for the other indices $\ell \neq i, \ell \in [1,M]$, $[\mathbf{m}(t+1)^\top L_\ell]_j - [\mathbf{m}(t)]_j \leq 0$. This amounts to assuming:

$$[\mathbf{m}(t)]_j := \max_{i\in[1,M]} [\mathbf{m}(t+1)^\top L_i]_j, \ \forall\ j \in [1,N].$$

Therefore also in this case the optimal input is the one that annihilates all the terms

$$\mathbf{w}(t)^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t), \qquad t \in [0, T-1],$$

where

$$\mathbf{w}(t)^\top = \mathbf{m}(t+1)^\top L - [\,\mathbf{m}(t)^\top \quad \mathbf{m}(t)^\top \quad \ldots \quad \mathbf{m}(t)^\top\,].$$

To illustrate this revised technique, we consider the same example (Example 8) addressed at the end of [11].

Example 2: Consider the BCN (2) and suppose that $N = 4$, $M = 4$ and

$$\begin{aligned}
L_1 &:= L \ltimes \delta_4^1 = [\,\delta_4^2 \quad \delta_4^3 \quad \delta_4^3 \quad \delta_4^4\,], \\
L_2 &:= L \ltimes \delta_4^2 = [\,\delta_4^2 \quad \delta_4^1 \quad \delta_4^3 \quad \delta_4^4\,], \\
L_3 &:= L \ltimes \delta_4^3 = [\,\delta_4^1 \quad \delta_4^3 \quad \delta_4^2 \quad \delta_4^4\,], \\
L_4 &:= L \ltimes \delta_4^4 = [\,\delta_4^2 \quad \delta_4^2 \quad \delta_4^4 \quad \delta_4^3\,].
\end{aligned}$$

We consider the problem of minimizing the cost function (6) for $T = 3$, by assuming

$$\mathbf{c}_f^\top = [\,1 \quad 0 \quad 0 \quad 0\,], \qquad \mathbf{c}^\top = 0,$$

and initial condition $\mathbf{x}(0) = \delta_4^4$.

We proceed according to the revised algorithm in order to maximize the function $\mathbf{c}_f^\top \mathbf{x}(3)$:

- $\mathbf{m}(3) = \mathbf{c}_f^\top = [\,1 \quad 0 \quad 0 \quad 0\,]$;
- $\mathbf{m}(2) = [\,1 \quad 1 \quad 0 \quad 0\,]$ and $K(2) = [\,\delta_4^3 \quad \delta_4^2 \quad \delta_4^1 \quad \delta_4^1\,]$;
- $\mathbf{m}(1) = [\,1 \quad 1 \quad 1 \quad 0\,]$ and $K(1) = [\,\delta_4^2 \quad \delta_4^2 \quad \delta_4^3 \quad \delta_4^1\,]$;
- $\mathbf{m}(0) = [\,1 \quad 1 \quad 1 \quad 1\,]$ and $K(0) = [\,\delta_4^2 \quad \delta_4^2 \quad \delta_4^2 \quad \delta_4^4\,]$.

As a consequence,

$$J^* = \max_{\mathbf{u}(\cdot)} J(\delta_4^4, \mathbf{u}(\cdot)) = \mathbf{m}(0)^\top \delta_4^4 = 1.$$

An optimal input sequence is

$$\mathbf{u}^*(0) = \delta_4^4, \quad \mathbf{u}^*(1) = \delta_4^3, \quad \mathbf{u}^*(2) = \delta_4^2,$$

and it corresponds to the state-trajectory

$$\mathbf{x}^*(0) = \delta_4^4, \quad \mathbf{x}^*(1) = \delta_4^3, \quad \mathbf{x}^*(2) = \delta_4^2, \quad \mathbf{x}^*(3) = \delta_4^1.$$

E. Constraints on the final state or forbidden states/transitions

The finite-horizon optimal control problem for the BCN (2) can be further enriched by introducing either constraints on the final state to be reached or on the states and/or transitions we want to avoid. Specifically, we may be interested in imposing a specific value on the final state $\mathbf{x}(T)$. If so, we can just use the previous set-up, by assuming that $\mathbf{c}_f$ has all entries equal to $+\infty$, except for the one corresponding to the final state we want to achieve (i.e., the jth entry, if we want that $\mathbf{x}(T) = \delta_N^j$). Similarly, if we want to avoid a certain state (say δ_N^j) or a certain transition (by this meaning the use of a specific input $\mathbf{u}(t) = \delta_M^i$ corresponding to a given state $\mathbf{x}(t) = \delta_N^j$), we just need to set to $+\infty$ some entries of the vector $\mathbf{c}$ (specifically, $[\mathbf{c}_i]_j = 0$ for all $i \in [1, M]$ in the former case, and $[\mathbf{c}_i]_j = 0$ for the given values of i and j in the latter case).

V. Optimal control of BCNs with penalty on the switching

The interpretation of a BCN as a Boolean switched system suggests a generalization of the cost function (6) we have considered up to now. Indeed, in addition to the cost on the state, weighted by the value of the input sample we apply at the same time instant, and hence depending on the specific subsystem we are using, we may want to penalize switchings, namely changes in the input value, meanwhile attributing zero cost to conservative inputs (namely to the case when $\mathbf{u}(t)$ coincides with $\mathbf{u}(t-1)$). This idea is formalized by the following cost function:

$$\begin{aligned} J(\mathbf{x}_0, \mathbf{u}(-1), \mathbf{u}(\cdot)) &= \mathbf{c}_f^\top \mathbf{x}(T) + \sum_{t=0}^{T-1} \mathbf{c}^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t) \\ &+ \sum_{t=0}^{T-1} \mathbf{p}^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{u}(t-1), \end{aligned} \tag{11}$$

where $\mathbf{c}_f \in \mathbb{R}^N$, $\mathbf{c} \in \mathbb{R}^{NM}$ and $\mathbf{p} \in \mathbb{R}^{M^2}$. To penalize switchings and attribute no penalties when the input does not change, it is sufficient to choose $\mathbf{p}$ in such a way that $\mathbf{p} \ltimes \delta_M^i \ltimes \delta_M^j$ is zero if $i = j$ and positive otherwise. Clearly different switchings may be penalized in different ways.

First of all, we may notice that in this set-up the initial condition is not only the state at $t = 0$, but also the input value at time $t = -1$. This suggests a way to tackle this problem, namely by introducing an augmented state variable:

$$\xi(t) := \mathbf{x}(t) \ltimes \mathbf{u}(t-1),$$

with known initial condition $\xi(0) = \mathbf{x}(0) \ltimes \mathbf{u}(-1)$. As a first goal, we want to derive the one-step updating law of the variable $\xi(t)$. To this end, we introduce some notation: assume that each matrix $L_i = L \ltimes \delta_M^i$ can be written as follows,

$$L_i = [\, L_{i1} \quad L_{i2} \quad \dots \quad L_{iN} \,], \qquad L_{ij} \in \mathcal{L}_N, \forall\ j \in [1, N].$$

Note that $L_{ij} = L \ltimes \delta_M^i \ltimes \delta_N^j$. It is not difficult to verify that

$$\xi(t+1) = \tilde{L} \ltimes \mathbf{u}(t) \ltimes \xi(t),$$

where

$$\begin{aligned} \tilde{L} = [\, & L_{11} \ltimes \delta_M^1 \mathbf{1}_M^\top \quad \dots \quad L_{1N} \ltimes \delta_M^1 \mathbf{1}_M^\top \quad | \\ & L_{21} \ltimes \delta_M^2 \mathbf{1}_M^\top \quad \dots \quad L_{2N} \ltimes \delta_M^2 \mathbf{1}_M^\top \quad | \quad \dots \\ & L_{M1} \ltimes \delta_M^M \mathbf{1}_M^\top \quad \dots \quad L_{MN} \ltimes \delta_M^M \mathbf{1}_M^\top \,]. \end{aligned}$$

We now want to prove that also the cost function (11) can be rewritten as in (6), by referring to the state variable $\xi(t)$. Indeed, we can easily verify that

$$\begin{aligned}\mathbf{c}_f^\top \mathbf{x}(T) &= [\,[\mathbf{c}_f]_1\mathbf{1}_M^\top \quad [\mathbf{c}_f]_2\mathbf{1}_M^\top \quad \dots \quad [\mathbf{c}_f]_N\mathbf{1}_M^\top\,]\,\xi(T)\\ &= (\mathbf{c}_f^\top \otimes \mathbf{1}_M^\top)\xi(T).\end{aligned}$$

On the other hand,

$$\begin{aligned}\mathbf{c}^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{x}(t) &= [\,[\mathbf{c}_1]_1\mathbf{1}_M^\top \quad \dots \quad [\mathbf{c}_1]_N\mathbf{1}_M^\top \quad |\\ &\qquad [\mathbf{c}_2]_1\mathbf{1}_M^\top \quad \dots \quad [\mathbf{c}_2]_N\mathbf{1}_M^\top \quad | \quad \dots\\ &\qquad [\mathbf{c}_M]_1\mathbf{1}_M^\top \quad \dots \quad [\mathbf{c}_M]_N\mathbf{1}_M^\top]\\ &\ltimes \quad \mathbf{u}(t) \ltimes \xi(t)\\ &= (\mathbf{c}^\top \otimes \mathbf{1}_M^\top) \ltimes \mathbf{u}(t) \ltimes \xi(t).\end{aligned}$$

Finally, if

$$\mathbf{p}^\top = [\,\mathbf{p}_1^\top \quad \mathbf{p}_2^\top \quad \dots \quad \mathbf{p}_M^\top\,],$$

where $\mathbf{p}_i^\top := \mathbf{p}^\top \ltimes \delta_M^i, i \in [1, M]$, then

$$\begin{aligned}&\mathbf{p}^\top \ltimes \mathbf{u}(t) \ltimes \mathbf{u}(t-1)\\ =\ &[\,\mathbf{p}_1^\top \quad \dots \quad \mathbf{p}_1^\top \quad |\,\dots|\, \quad \mathbf{p}_M^\top \quad \dots \quad \mathbf{p}_M^\top\,] \ltimes \mathbf{u}(t) \ltimes \xi(t).\end{aligned}$$

This implies that the cost function (11) can be rewritten as

$$J(\xi(0), \mathbf{u}(\cdot)) = \tilde{\mathbf{c}}_f^\top \xi(T) + \sum_{t=0}^{T-1} \tilde{\mathbf{c}}^\top \ltimes \mathbf{u}(t) \ltimes \xi(t),$$

where

$$\begin{aligned}\tilde{\mathbf{c}}_f^\top &:= \mathbf{c}_f^\top \otimes \mathbf{1}_M^\top,\\ \tilde{\mathbf{c}}^\top &:= \mathbf{c}^\top \otimes \mathbf{1}_M^\top\\ &\quad + [\,\mathbf{p}_1^\top \quad \dots \quad \mathbf{p}_1^\top \quad | \quad \dots \quad | \quad \mathbf{p}_M^\top \quad \dots \quad \mathbf{p}_M^\top\,].\end{aligned}$$

Consequently, the approach developed in section III may be successfully applied to obtain the optimal solution also in this case.

REFERENCES

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

[2] D. Cheng and J.B. Liu. Stabilization of Boolean control networks. In *Proceedings of the Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference*, pages 5269–5274, Shanghai, China, 2009.

[3] D. Cheng and H. Qi. Linear representation of dynamics of Boolean Networks. *IEEE Trans. Automatic Control*, 55, (10):2251 – 2258, 2010.

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

[5] D. Cheng, H. Qi, and Z. Li. *Analysis and control of Boolean networks*. Springer-Verlag, London, 2011.

[6] E. Fornasini and M. E. Valcher. Observability, reconstructibility and state observers of Boolean control networks. *IEEE Tran. Aut. Contr.*, 2013. to appear.

[7] E. Fornasini and M. E. Valcher. On the periodic trajectories of Boolean Control Networks. *Automatica*, 2013. to appear.

[8] E. Fornasini and M.E. Valcher. Observability and reconstructibility of Boolean control networks. In *Proceedings of the 51st Conference on Decision and Control (CDC2012)*, pages 2574–2580, Maui, Hawaii, 2012.

[9] D. G. Green, T. G. Leishman, and S. Sadedin. The emergence of social consensus in Boolean networks. In *Proc. IEEE Symp. Artificial Life (ALIFE07)*, pages 402–408, Honolulu, HI, 2007.

[10] S.A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theoretical Biology*, 22:437467, 1969.

[11] D. Laschov and Margaliot M. A Pontryagin maximum principle for multi-input boolean control networks. In K. Kaslik and S. Sivasundaram, editors, *Recent Advances in Dynamics and Control of Neural Networks*, to appear, 2013.

[12] D. Laschov and M. Margaliot. A maximum principle for single-input Boolean Control Networks. *IEEE Trans. Automatic Control*, 56, no. 4:913–917, 2011.

[13] D. Laschov and M. Margaliot. Controllability of Boolean control networks via the Perron-Frobenius theory. *Automatica*, 48:1218–1223, 2012.

[14] H. Li and Y. Wang. Boolean derivative calculation with application to fault detection of combinational circuits via the semi-tensor product method. *Automatica*, 48, (4):688–693, 2012.

[15] Y. Zhao, Z. Li, and D. Cheng. Optimal control of logical control networks. *IEEE Trans. Automatic Control*, 56, (8):1766–1776, 2011.

[16] Q. Zhu and G. Xie. Finite-horizon optimal control of discrete-time switched linear systems. *Mathematical Problems in Engineering*, Article ID 483568:doi:10.1155/2012/483568, 2012.