Computational method for the inference of therapeutic targets and sequence of treatments

Jérémie Pardo

université
PARIS-SACLAY

Méthodes d'inférence de cibles thérapeutiques et de séquences de traitement

Computational methods for the inference of therapeutic targets and sequences of treatment

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°580,Sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Informatique
Graduate School : Informatique et science du numérique
Référent : Université d'Evry val d'essone

Thèse préparée dans l'unité de recherche Université Paris-Saclay, Univ Evry, IBISC, 91020, Evry-Courcouronnes, France, sous la direction de Franck Delaplace, Professeur à l'Université de Paris-Saclay - Univ. Evry, et de le co-encadrement de Sergiu IVANOV, maître de conférences à l'Université de Paris-Saclay - Univ. Evry

Thèse soutenue à Évry-Courcouronnes, le 3 février 2022, par

Jérémie PARDO

Composition du jury

Philippe Dague Professeur émérite, Université Paris-Saclay	Examinateur
François Fages Directeur de rechercher, INRIA Paris-Saclay (DR1)	Examinateur
Loïc Paulevé Chargé de recherche, CNRS (LaBRI)	Rapporteur & Examinateur
Elisaveth Remy Directrice de rechercher, CNRS (I2M)	Présidente
Adrien Richard Chargé de recherche, CNRS (I3S)	Examinateur
Syvain Sené Professeur, Université d'Aix Marseille (LIS)	Rapporteur & Examinateur
Franck Delaplace Professeur, Université Paris-Saclay - Univ. Evry (IBISC)	Directeur de thèse
Sergiu Ivanov Maître de conférences, Université Paris-Saclay - Univ. Evry (IBISC)	Co-encadrant de thèse

THESE DE DOCTORAT
NNT : 2022UPASG011

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Méthodes d'inférence de cibles thérapeutiques et de séquences de traitement
Mots clés: Réseaux booléens contrôlés - Séquence de contrôle - Médecine des réseaux

Résumé: Un enjeu majeur de la médecine des réseaux est l'identification des perturbations moléculaires induites par les maladies complexes et les thérapies afin de réaliser une reprogrammation cellulaire. L'action de la reprogrammation est le résultat de l'application d'un contrôle. Dans cette thèse, nous étendons le contrôle unique des réseaux biologiques en étudiant le contrôle séquentiel des réseaux booléens. Nous présentons un nouveau cadre théorique pour l'étude formelle des séquences de contrôle. Nous considérons le contrôle par gel de noeuds. Ainsi, une variable du réseau booléen peut être fixée à la valeur 0, 1 ou décontrôlée. Nous définissons un modèle de dynamique contrôlée pour le mode de mise à jour synchrone où la modification de contrôle ne se produit que sur un état stable. Nous appelons CoFaSe le problème d'inférence consistant à trouver une séquence de contrôle modifiant la dynamique pour évoluer vers une propriété ou un état souhaité. Les réseaux auxquels sera appliqué CoFaSe auront toujours un ensemble de variables incontrôlables. Nous montrons que ce problème est PSPACE-dur. L'étude des caractéristiques dynamiques du problème CoFaSe nous a permis de constater que les propriétés dynamiques qui impliquent la nécessité d'une séquence de contrôle émergent des fonctions de mise à jour des variables incontrôlables. Nous trouvons que la longueur d'une séquence de contrôle minimale ne peut pas être supérieure à deux fois le nombre de profils des variables incontrôlables. À partir de ce résultat, nous avons construit deux algorithmes inférant des séquences de contrôle minimales sous la dynamique synchrone. Enfin, l'étude des interdépendances entre le contrôle séquentiel et la topologie du graphe d'interaction du réseau booléen nous a permis de découvrir des relations existantes entre structure et contrôle. Celles-ci mettent en évidence une borne maximale plus resserrée pour certaines topologies que celles obtenues par l'étude de la dynamique. L'étude sur la topologie met en lumière l'importance de la présence de cycles non-négatifs dans le graphe d'interaction pour l'émergence de séquences minimales de contrôle de taille supérieure ou égale à deux.

Title: Computational methods for the inference of therapeutic targets and sequences of treatment
Keywords: Boolean control network - Control sequence - Network medicine

Abstract: Network controllability is a major challenge in network medicine. It consists in finding a way to rewire molecular networks to reprogram the cell fate. The reprogramming action is typically represented as the action of a control. In this thesis, we extended the single control action method by investigating the sequential control of Boolean networks. We present a theoretical framework for the formal study of control sequences.We consider freeze controls, under which the variables can only be frozen to 0, 1 or unfrozen. We define a model of controlled dynamics where the modification of the control only occurs at a stable state in the synchronous update mode. We refer to the inference problem of finding a control sequence modifying the dynamics to evolve towards a desired state or property as CoFaSe. Under this problem, a set of variables are uncontrollable. We prove that this problem is PSPACE-hard. We know from the complexity of CoFaSe that finding a minimal sequence of control by exhaustively exploring all possible control sequences is not practically tractable. By studying the dynamical properties of the CoFaSe problem, we found that the dynamical properties that imply the necessity of a sequence of control emerge from the update functions of uncontrollable variables. We found that the length of a minimal control sequence cannot be larger than twice the number of profiles of uncontrollable variables. From this result, we built two algorithms inferring minimal control sequences under synchronous dynamics. Finally, the study of the interdependencies between sequential control and the topology of the interaction graph of the Boolean network allowed us to investigate the causal relationships that exist between structure and control. Furthermore, accounting for the topological properties of the network gives additional tools for tightening the upper bounds on sequence length. This work sheds light on the key importance of non-negative cycles in the interaction graph for the emergence of minimal sequences of control of size greater than or equal to two.

Remerciements

Cette thèse a été un travail long et sinueux. L'ensemble des résultats scientifiques qui m'ont permis d'obtenir le grade de docteur n'aurait jamais pu exister sans la présence de mon directeur de thèse Franck Delaplace et la présence de mon co-encadrant Sergiu Ivanov. Je vous suis très reconnaissant pour toute l'aide que vous m'avez apportée durant ces quelques années de collaboration.

Merci Franck pour nos échanges parfois tumultueux qui ont souvent permis de démêler mes idées et de recentrer le travail sur les points essentiels. Tes conseils sur mes présentations orales m'ont toujours été d'une aide précieuse. Surtout, merci d'avoir cru en moi et de m'avoir encouragé à faire cette thèse. Réaliser une thèse sous ta direction fut un grand plaisir.

Merci Sergiu pour tout le soutien dont tu m'as fait part et nos nombreuses conversations scientifiques. Ta présence a souvent été un point d'équilibre pendant nos réunions de recherche à trois. Je te remercie énormément pour le temps que tu as investi dans mon manuscrit et correction de mes preuves. Être ta première expérience de direction de thèse fut un grand plaisir.

Je tiens à remercier Élisaveth Remy d'avoir présidé le jury de ma thèse. Je tiens également à remercier les membres du jury Philippe Dague, François Fages et Adrien Richard d'avoir accepté de faire partie de ce jury et pour leurs questions pertinentes posées lors de ma soutenance. Je souhaite remercier plus particulièrement Sylvain Sené et Loïc Paulevé qui ont accepté de rapporter ma thèse. Merci pour leur relecture particulièrement minutieuse et l'ensemble de leurs commentaires pertinents qui m'ont permis d'améliorer grandement la qualité de ce manuscrit.

À tous mes collègues membres permanents, secrétaires, stagiaires, doctorants actuels ou anciens du laboratoire IBISC qui m'ont accueilli parmi eux, je vous remercie pour tout ce bon temps que l'on aura passé ensemble.J'ai passé de super moments de détente et de collaboration avec vous. J'en garde de nombreux et agréables souvenirs.

Je souhaite aussi remercier mes amies qui ont réalisé le déplacement à ma soutenance de thèse. Vous voir a été un réel plaisir.

Pour finir, je souhaite remercier ma famille qui a été présente pour moi et qui m'a soutenu et m'a supporté lors de cette longue aventure qu'a été ma thèse.

Contents

1 - Introduction

Cell reprogramming consists in the control of molecular processes and modifying gene expression to induce a particular cell behaviour naturally or artificially. The potential outcomes of reprogramming have valuable benefits regarding the essential challenges of health: cancerous targeted therapy, complex disease aetiology, regenerative medicine, stem cells monitoring, etc. [33]. Despite the impressive progress in cell reprogramming during the past decade, more breakthroughs are required before cellular reprogramming yields routine clinical use [59]. The main issues lie in the discovery of reliable ways to trigger the reprogramming process and to understand exactly how its mechanisms work. In this endeavour, the definitions of suitable theoretical frameworks and computational methods are crucial for enabling the analysis and the design of the *reprogramming patterns* responsible for the phenotypic switch.

Finding therapeutic targets is based on the study of molecular models of disease. These models are frameworks that define the causal relationship between disturbances at the molecular level and the diseased or healthy forms of an organism. There are two complementary approaches for studying diseases at the molecular level: empiricism and rationalism [29, 60].

The empirical method involves establishing statistical measures between sets of patients' symptoms and their molecular characteristics. The rational method relies on developing mechanistic models that theorise the functioning of an organism and explain the emergence of the symptoms of the disease via disturbances in the model. Therefore, the discovery of targeted therapies can be based on high-throughput screening, which tests, for example, the effectiveness of many molecules actions without knowing their mode of action or predicts drug effects based on biological models [56].

Whether empirical or rational methodology, developments in precision medicine need computer science [14, 27]. The empirical methods suppose the ability to analyse considerable masses of data resulting from the progress of molecular biology. The rational methods imply the development of disease models and algorithms for obtaining the desired predictions.

In this manuscript, we focus on a rational approach based on the study of biological models. In [62], the authors relate mutations to their network effects:

nonsense mutation, out-of-frame insertion or deletion and defective splicing are interpreted as node or arc deletions, whereas missense mutation and in-frame insertion or deletion can be modelled as node or arc additions. Moreover, in [15], the authors classify mutations according to how they affect signalling networks and distinguish between mutations that constitutively activate or inhibit enzymes (nodes) and mutations that rewire the interactions (arcs). Similarly, [16] interpret targeted therapies as network rewiring. The effects of mutations and drugs can, thus, be described as elementary topological actions on the network: the deletion or insertion of nodes and arcs. *Cell reprogramming is then viewed as network alteration based on these topological actions.* The impact of the actions on the network should be evaluated from a model of dynamics translating the topological actions into dynamical alteration of the trajectories. Accordingly, phenotypic changes are assessed at the molecular level via the measurement of the state of particular molecules called *biomarkers*—observable indicators of biological processes whose molecular signature variation discriminates the phenotypes [16, 55]. The signatures must be observed over a significant period of time to testify their relevance, and thus are assumed to be met concomitantly with a stability condition of the biological system.

This approach is part of *network medicine* [6],which aims to address drug target discovery and the elucidation of disease mechanisms through network analysis by renewing the phenotype-genotype relationship into the association of a phenotype to some network perturbations [53]. Albert-László Barabási states that biological systems contain many components that are connected in complicated relationships but are organised by simple principles. Using network theory, the organising principles can be comprehensively analysed by representing systems as complex networks. Network medicine is based on the idea that understanding the complexity of gene regulation, metabolic reactions, and protein–protein interactions, as well as using a network representation of those processes, will shed light on the causes and mechanisms of diseases.This thesis is, therefore, rooted in the field of network medicine.

We choose to use Boolean networks as a network representation for the cellular processes. Boolean networks are widely used to model biological systems in network medicine. These networks are discrete dynamics systems introduced in biology by McCulloch and Pitts [38] as a model for the transmission of information between neurons, and by Stuart Kauffman [28] and Rene Thomas [57] to model gene regulatory networks. The molecules of the system are represented as Boolean variables. By convention, the value 1 corresponds to an active or present state, and 0 to an inactive or absent state. Although Boolean networks are a rough

simplification of genetic reality, as genes are not simple binary switches, they can correctly capture many expression patterns of genes [2, 10].

Recent research in computational biology has provided novel inference methods for reprogramming a system to make its dynamics converge towards an expected fate. These works use *Boolean control networks* (BCNs). Its a model specifying the actions as controls on Boolean network. Various approaches have been proposed, such as stuck-at fault models and SAT-based methods [32], motif-related heuristics [61], algebraic approaches [39], and abductive methods [8]. These works have been validated using real biological cases showing their adequacy for drug therapy prediction. The state of art related to Boolean networks reveals that the methods are currently focused on computing a single network action modelled as a control input to reprogramme the dynamics in order to reach stable states that meet some expected properties assessed at the molecular level.

However, more complex schemes may require a control sequence. For example, in some biological cases, a sequence of mutations is observed, or a therapy involves a scheduled protocol for administering drugs. Typically, tumorigenesis results from a multi-step process governed by sequential genetic alterations. A colorectal tumour offers a paradigmatic system illustrating this sequential progression. Fearon and Vogelstein demonstrated that colorectal cancer tumorigenesis relies on a sequence of disturbances of three genes [22], called the 'Vogelstein sequence'. This sequence suggests that acquiring of a cancerous phenotype requires undergoing various intermediary stages and that, if these same disturbances occur in a different order from that observed, the cell dies before developing its cancerous potential.

Furthermore, in [31], the authors describe a systematic approach to identifying efficient drug combinations in killing cancer cells depending on changes in the order and duration of drug exposure. They found that some drug combinations (EGFR inhibitor) can synergise the apoptotic response to DNA-damaging chemotherapy for a subset of triple-negative breast cancers if the drugs are administered sequentially but not simultaneously, leading to an appropriate dynamics rewiring of oncogenic signalling networks.

Therefore, the study of sequential control is a natural extension of the previous works. Investigating control sequences can, as a long-term perspective, possibly explain the causes of diseases via sequences of perturbations and help discover therapeutic regimens.

The subject of control sequence inference is vast and difficult. Related ques-

tions to this problem could find their foundation in the problems related to the control of systems. We can cite, for example the control theory for linear systems, in which control system design addresses the problem of making a concrete physical system behave according to certain desired specifications by using a device called a controller [58].We can also consider the study of the robustness of the network and its boundary conditions, which offers insights into the behaviour of interacting systems and can provide important leads for finding the desired controls [19, 17]. In this manuscript, we only consider sequential control on Boolean networks, which is an under-researched area. The main studies on this subject are those of Mandon et al. , who researched the temporal reprogramming of Boolean networks in [35, 37, 36]. Given a trajectory, they identified the appropriate states at which a control should be applied, and deduced the corresponding controls (perturbations) to reach an expected state. The authors objective was minimising the total number of perturbations, particularly regarding known one-step controls. These series of publications emphasiSe the computational complexity of sequence inference.

In this thesis, we are interested in the definition of a framework and computational methods for the inference of control sequences for Boolean networks. The problem we pose is as follows:

If not all variables are controllable, how can we find a minimal control sequence modifying the dynamics to evolve towards a desired state from an initial state if such a state is not reachable with a one-step control?

In the model of controlled dynamics that we propose, we consider the constraints of its ultimate application in biology. We impose that objectives are determined on stable states and that the modification of the control only occurs at a stable state. Studying the dynamical characteristics (Chapter 5) of this framework enabled us to bound the length of control sequences of minimal size. Furthermore, studying the causality of sequential controls from the viewpoint of the interaction graph (Chapter 7) offered key insights to reduce such bounds. This work helps reveal the key importance of non-negative cycles in the interaction graph regarding the emergence of minimal sequences of control of size greater than or equal to two.

This thesis is structured as follows: In Chapter 2, we present the Boolean network model, its model of dynamics, and its interaction graph. We also recall the formalism of the Boolean control network which is a function generating Boolean networks according to control parameters. These control parameters enable us to

model structural disturbances of the Boolean network interaction graph. In this manuscript, we particularly focus on the control of Boolean network nodes. More precisely on control input that freezes nodes at a specific value.

In Chapter 3, we formalise the controlled dynamics that extend the Boolean network dynamics by revealing how the system evolves through a sequence of control inputs. We explain the inference problem regarding finding a control sequence that modifies the dynamics to evolve towards a desired state or property. This problem is denoted as CoFaSe (*i.e.*, *Controlled Fate in Sequence*). Finally, we prove that the inference of a control sequence satisfying CoFaSe is PSPACE-hard.

In Chapter 4, we present in greater detail the state of art regarding the control of Boolean networks. In particular, we discuss the similarities and differences to the framework proposed by Mandon et al. in [35, 37, 36].

In Chapter 5, we describe the dynamical properties of the sequential inference problem. By partitioning the network variables into a set of controllable and uncontrollable variables, we obtain concrete bounds regarding the length of minimal control sequences. These upper bounds reduce the potential solution space of the inference problem.

In Chapter 6, we propose two computational approaches for inferring minimal sequences of control. The first approach always finds a minimal sequence of control if such a sequence exists. The second approach may not always find a minimal sequence of control but generally requires less computation time. In the final section of this chapter, we benchmark and discuss the performance of the two algorithms.

In Chapter 7, we study the relationship between sequential control and the topology of the interaction graph of the Boolean network. Accounting for the topological properties of the network provides additional tools for tightening the upper bounds regarding sequence length. This approach enables us to study the causal relationships between structure and control.

2 - Boolean control network

In this manuscript, we will formalise the Boolean network reprogramming by relying on Boolean control network. A BCN extends the Boolean network by adding Boolean controls. The reprogramming of Boolean networks leads to the modification of their dynamics. We more specifically focus on a particular class of control called the *freezing control* , in which a *control input* definitively freezes a variable state to a specific value.

In this chapter, we first recall the main definitions of Boolean networks (Section 2.1) and their interaction graph (Section 2.2). Then, we define the extension to BCNs (Section 2.3).

2.1 . Boolean network

Boolean function. A Boolean function is defined as $f : \mathbb{B}^n \to \mathbb{B}$ where $\mathbb{B} = \{0, 1\}$ is the Boolean domain and n is its number of arguments. Every Boolean function can be expressed as a propositional formula with n variables $x_1, \ldots, x_n$. For example, $f(x_1, x_2, x_3) = (x_1 \wedge x_3) \vee (\neg x_1 \wedge x_2)$ is a Boolean formula which have for variables x_1, x_2 and x_3.

Boolean network. A Boolean network is a discrete dynamical system defined on a finite number of Boolean variables $X = \{x_1, \ldots, x_n\}$. The network is defined by a collection of Boolean functions,

$$F : \mathbb{B}^n \to \mathbb{B}^n,$$
$$F = \{x_i = f_i(x_1, \ldots, x_n) \mid 1 \leq i \leq n\},$$

in which each f_i is a propositional formula computing the instantiation of x_i. An example of a Boolean network is presented in Figure 2.1.

$$F = \begin{cases} x_1 = (x_1 \wedge \neg x_2) \vee (x_1 \wedge \neg x_3) \vee (\neg x_1 \wedge x_2 \wedge x_3) \\ x_2 = (x_1 \wedge x_3) \vee (\neg x_1 \wedge x_2) \\ x_3 = (x_1 \wedge x_3) \vee (\neg x_1 \wedge \neg x_2) \end{cases}$$

Legend: A Boolean network F with three variables x_1, x_2 and x_3.

Figure 2.1: A Boolean network.

State. A state s belonging to the set of states S_X is an interpretation assigning a Boolean value to the variables (*i.e.*, $s : X \to \mathbb{B}$). The evolution of each variable x_i depends on its update Boolean function f_i and a state s. Table 2.1 contains the result of applying the set of Boolean functions of the Boolean network in Figure 2.1 to each state $s \in S_X$.

$s_{\{x_1,x_2,x_3\}}$	$f_1(s)$	$f_2(s)$	$f_3(s)$
000	0	0	1
001	0	0	1
010	0	1	0
011	1	1	0
100	1	0	0
101	1	1	1
110	1	0	0
111	0	1	1

Legend: Application of the set of Boolean functions of the Boolean network F of Figure 2.1 to its set of states S_X.

Table 2.1: Application of the Boolean functions of a Boolean network.

Model of dynamics. The model of dynamics describes the evolution of states for all variables via a labelled transition system $\langle \longrightarrow, M, S_X \rangle$, in which the states are updated according to an updating policy $M \subseteq \mathbf{2}^X$ called the *mode*, which is a cover of X ($\bigcup_{m \in M} m = X$).

$f_{\downarrow X'}$ defines the restriction/projection of the function to X', such that f is only defined for the elements of $X' \subset X$.

Each transition relation ($\longrightarrow \subseteq S_X \times M \times S_X$) is labelled by the set of updated variables m:

$$s \xrightarrow{m} s' \stackrel{\text{def}}{=\!=} s' = (F_{\downarrow m}(s) \cup s_{X \setminus m}).$$

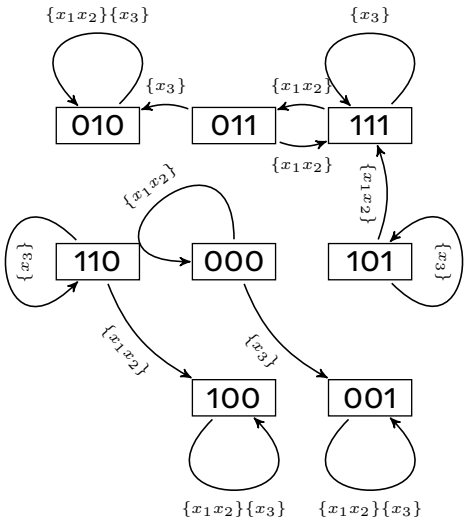
The global transition relation is defined as: $\longrightarrow = \bigcup_{m \in M} \xrightarrow{m}$. A path[1] $s \longrightarrow^* s'$ characterises a trajectory from s to s'.

For example, if we consider the Boolean network of Figure 2.1 and the updating policy $M = \{\{x_1, x_2\}, \{x_3\}\}$. The updating policy M means that from the state 011, it is possible to carry out one of the two following transitions:

[1] $\longrightarrow^*$ is the reflexive and transitive closure of the transition relation.

- One transition in which the variables x_1 and x_2 are updated and the state 111 is reached.
- One transition in which the variable x_3 is updated and the state 010 is reached.

The updating policy M provides us with the following states graph:



Legend: Dynamics of the Boolean network F of Figure 2.1 under the updating policy $M = \{\{x_1, x_2\}, \{x_3\}\}$. Arcs are labelled by the sets of updated variables realizing the transitions.

Figure 2.2: Update mode of a Boolean network.

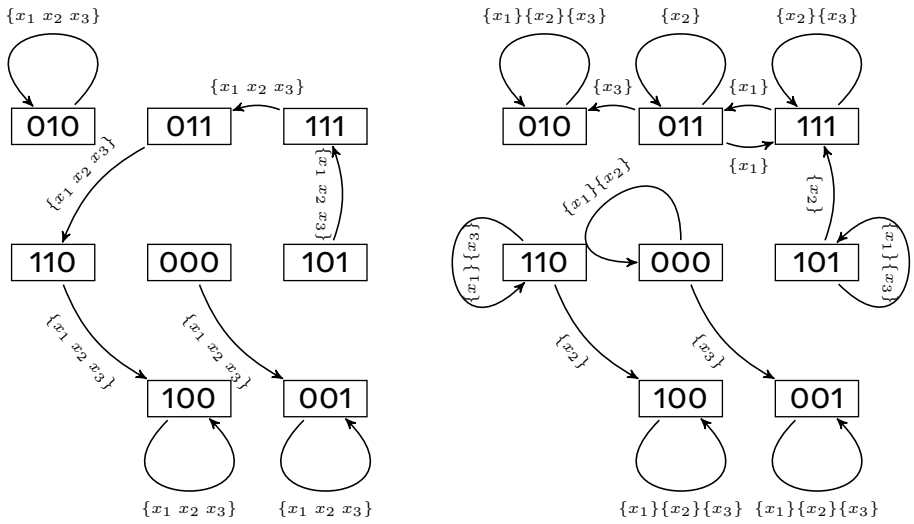
In biological modelling, various updating modes are proposed to model the numerous constraints of the abstracted system. There is no consensus regarding which updating mode for Boolean networks is the most representative of the biological reality. Thus, the choice of updating mode strongly depends on the nature of the studied problem. Indeed, each updating mode has distinct features and can have a wide range of effects on the dynamics of the Boolean network.

In this manuscript, we only introduce two of the most used updating modes: the *synchronous* mode and the *asynchronous* mode. These two modes only represent a small part of the spectrum of update modes introduced in the literature. For example, the updating functions may be composed, as in block-sequential [50] and block-parallel [20] updating modes. Other updating modes may also make use of parameters that at first sight cannot be directly captured by these deterministic updates [46]. Memory Boolean networks [24, 25] and Interval Boolean networks [12] respectively consider delay and duration and Most Permissive Boolean

networks [44] consider thresholds in their updating functions.

In the *synchronous* mode, all the variables are updated during a transition ($M = \{X\}$). The synchronous update mode is *deterministic* since all transitions are functions.[2]

By contrast, in the *asynchronous* mode, only one variable is updated per transition ($M = \{\{x_i\}\}_{x_i \in X}$). The asynchronous update mode is *non-deterministic* since its transitions are ordinary relations (not functions). Figure 2.3 shows the synchronous and asynchronous dynamics state graphs of the Boolean network of Figure 2.1.



Synchronous *Asynchronous*

Legend: Dynamics of the Boolean network F of Figure 2.1 under two update modes. Arcs are labelled by the sets of updated variables realizing the transitions. On the left is the dynamics of F under the synchronous update mode. On the right is the dynamics of F under the asynchronous update mode.

Figure 2.3: Synchronous and Asynchronous update modes of a Boolean network.

Equilibrium. An attractor $A \subseteq S_X$ is a set of states from which only states of A are reachable and in which all states of A are reachable from any state of A. A is an attractor if and only if:

[2] One result for each input value.

$$A \neq \emptyset \wedge \forall s \in A, \forall s' \in S_X \setminus A : \ \neg(s \longrightarrow^* s') \wedge \forall s, s' \in A : \ s \longrightarrow^* s'.$$

A *fixpoint attractor* or *stable state* s is a particular attractor whose cardinality, denoted $|A|$, is equal to 1. A state s is denoted as a stable state by the following notation:

$$\text{STBL}_F(s) \stackrel{\text{def}}{=\!=} \forall m \in M : \ s \xrightarrow{m} s.$$

A *cyclic attractor* is an attractor of size $|A| > 1$ that forms a cycle in the dynamics of the Boolean network. Note that under the synchronous update mode of a Boolean network, an attractor can only be a fixpoint or a cyclic attractor.

A *basin of attraction* of an attractor A is a set of states in which each of its states will always eventually reach a state of A. B is a basin of attraction of an attractor A if:

$$A \subseteq B \wedge \forall s \in B, \exists s' \in A : s \longrightarrow^* s'$$
$$\wedge \forall s \in B, \forall s' \in S_X : s \longrightarrow^* s' \implies s' \in B.$$

In the left graph of Figure 2.3, the states 010, 100 and 001 are stable states. Their largest basins of attraction are, respectively, $\{010\}$, $\{101, 111, 011, 110, 100\}$, and $\{000, 001\}$.

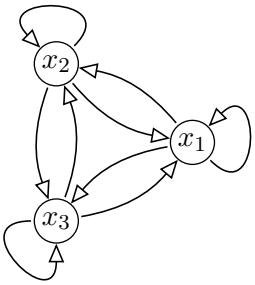
2.2 . Boolean networks interaction graph

Interaction graph. The interaction graph of a Boolean network captures the interdependence of the network variables in the dynamics. For a Boolean network F with the set of variables X, its interaction graph is a directed graph $\langle X, \longrightarrow\!\triangleright \rangle$, which contains the arc $x_i \longrightarrow\!\triangleright x_j$ if a change in x_i may leads to a change in x_j. Formally, an interaction is defined as follows:

$$x_i \longrightarrow\!\triangleright x_j \stackrel{\text{def}}{=\!=} \exists s, s' \in S_X : s_{x_i} \neq s'_{x_i} \wedge s_{X \setminus x_i} = s'_{X \setminus x_i} \wedge f_j(s) \neq f_j(s').$$

Figure 2.4 shows the interaction graph of the Boolean network of Figure 2.1.

Signed interaction graph. The signed interaction graph extends the notion of dependence by classifying arcs in three categories: a monotonic increase in x_j (sign 1), a monotonic decrease in x_j (sign -1), or a non-monotonic interaction



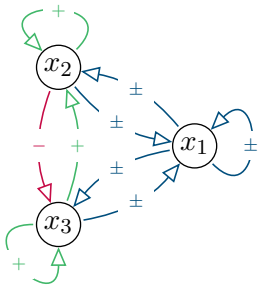
Legend: Interaction graph of the Boolean network F of Figure 2.1.

Figure 2.4: The interaction graph of a Boolean network.

depending on the state of the other variable (sign 0). The signs are respectively labelled by: $+$ for 1, $-$ for -1, and $\pm$ for 0. Formally, the signed interaction graph of a Boolean graph $\langle X, \longrightarrow\!\!\triangleright, \sigma \rangle$, in which the relation $x_i \longrightarrow\!\!\triangleright\, x_j$ is defined in the same way as for the interaction graph, and $\sigma : (\longrightarrow\!\!\triangleright) \to \{-1, 0, 1\}$ is the arc labelling function defined according to the following equations and graphically represented by $\{-, \pm, +\}$:

$$
\begin{aligned}
x_i \xrightarrow{+}\!\!\triangleright\, x_j \overset{\text{def}}{=\!=} x_i \longrightarrow\!\!\triangleright\, x_j \wedge \\
\forall s, s' \in S_X : s_{x_i} \leq s'_{x_i} \wedge s_{X \setminus x_i} = s'_{X \setminus x_i} \implies f_j(s) \leq f_j(s'), \\
x_i \xrightarrow{-}\!\!\triangleright\, x_j \overset{\text{def}}{=\!=} x_i \longrightarrow\!\!\triangleright\, x_j \wedge \\
\forall s, s' \in S_X : s_{x_i} \leq s'_{x_i} \wedge s_{X \setminus x_i} = s'_{X \setminus x_i} \implies f_j(s) \geq f_j(s').
\end{aligned}
$$

The arcs for which neither of the two equations is true, receive the label $\pm$. Figure 2.5 displays the signed interaction graph of the Boolean network of Figure 2.1.



Legend: Signed Interaction graph of the Boolean network F of Figure 2.1.

Figure 2.5: The signed interaction graph of a Boolean network.

The arcs labelled $+$ refer to *positive* arcs, whereas those labelled $-$ refer to *negative* arcs.

Cycle. In an interaction graph, a cycle is a finite sequence of distinct arcs in the same direction that joins a sequence of vertices in which the only repeated vertices are the first and last. In a signed interaction graph, the sign of a cycle is the product of the sign of its arcs.

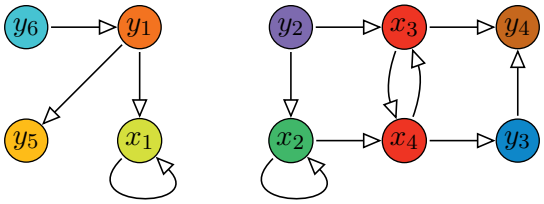
- A cycle is a *positive cycle* if the product of the signs is equal to 1 (*i.e.*, if and only if it contains an even number of '$-$' arcs and no '$\pm$' arcs).
- A cycle is a *negative cycle* if the product of the signs is equal to -1 (*i.e.*, if and only if it contains an odd number of '$-$' arcs and no '$\pm$' arcs).
- A cycle is a *positive/negative cycle* (positive and negative at the same time) if the product of the signs is equal to 0 (*i.e.*, if it contains at least a '$\pm$' arc). Depending on the instantiation of the variables of the network, such cycles can behave as positive or negative cycles.
- A cycle is a *non-negative cycle* if the product of the signs is *not* equal to -1 (*i.e.*, the cycle is a positive cycle or a positive/negative cycle).

For example, in the interaction graph of F in Figure 2.5, the cycles $\{x_2, x_2\}, \{x_3, x_3\}$, are positive cycles, the cycle $\{x_2, x_3, x_2\}$, is a negative cycle and the cycles $\{x_1, x_1\}, \{x_1, x_2, x_3, x_1\}$, are positive/negative cycles.

Strongly connected component. A subgraph of a directed graph is called a strongly connected component (SCC) if every vertex is reachable from every other vertex. We can distinguish between trivial SCCs, which do not contain cycles, and complex SCCs, which contain one or more cycles. Since an SCC with more than one variable necessarily possesses a cycle, all trivial SCCs are of cardinality 1. All signed interaction graphs can be partitioned into SCC modules. For an example of a partitioned interaction graph into SCC modules, see Figure 2.6.

In the rest of this manuscript, we only study complex strongly connected components. For the sake of clarity, complex SCCs are simply referred to as SCCs.

Upstream, downstream, and disconnected variables. For the sake of proof in the remainder of this manuscript, we now define the notion of *upstream, downstream, and disconnected variables in regards to a set of variables as follows*:



Legend: Interaction graph partitioned into SCC modules where each color corresponds to a distinct SCC. The variables x_1 to x_4 belong to complex SCCs in contrary to the variables y_1 to y_6 which belong to trivial SCCs.

Figure 2.6: Interaction graph partitioned into SCC modules

Let $\mathcal{A}$ be a set of variables in the interaction graph of the network F. The variables of F can be classified according to $\mathcal{A}$ into four sets:

- $\mathcal{A}$ the set of the reference variables.
- $\mathcal{A}^{\swarrow} = \{x \mid x \in X, x \notin \mathcal{A}, \exists x' \in \mathcal{A} : x \longrightarrow^* x'\}$ the set of variables upstream of the set of variables $\mathcal{A}$.
- $\mathcal{A}^{\searrow} = \{x \mid x \in X, x \notin \mathcal{A}, \exists x' \in \mathcal{A} : x' \longrightarrow^* x\}$ the set of variables downstream of the set of variables $\mathcal{A}$.
- $\mathcal{A}^{\times} = X \backslash (\mathcal{A} \cup \mathcal{A}^{\swarrow} \cup \mathcal{A}^{\searrow})$ the set of variables disconnected from the set of variables $\mathcal{A}$.

For example, in Figure 2.6, if we take the set of variables $\mathcal{A} = \{x_1\}$ we will have $\mathcal{A}^{\swarrow} = \{y_1, y_6\}$, $\mathcal{A}^{\searrow} = \emptyset$, and $\mathcal{A}^{\times} = \{x_2, x_3, x_4, y_2, y_3, y_4, y_5\}$, and if we take the set of variables $\mathcal{B} = \{x_3, y_4\}$, we will have $\mathcal{B}^{\swarrow} = \{x_2, x_4, y_2, y_3\}$, $\mathcal{B}^{\searrow} = \{x_4, y_3\}$ and $\mathcal{B}^{\times} = \{x_1, y_1, y_5, y_6\}$.

2.3 . Boolean control network

The Boolean control network (BCN) extends the Boolean network by adding controls on variables. A control is represented by an additional parameter whose state is set simultaneously and instantaneously when a control is applied.

More formally, a Boolean control network F_U is a function generating a Boolean network from an interpretation $\mu \in S_U$ of control parameters $U = \{u_1, \ldots, u_j, \ldots, u_m\}$, which is called a *control input*. This input is defined as follows:

$$F_U = \{x_i = f_i(x_1, \ldots, x_n, u_1, \ldots, u_m) \mid 1 \leq i \leq n\},$$

For each instantiation of control input μ, the Boolean control network F_U generates a Boolean network F_μ, modelling the application of a control on the initial Boolean network F.

The freezing control assigns a definite value to each variable. The two possible freezing outcomes, 0 or 1, are supported by two parameters with two distinct regimes: either they freeze the variable or remain idle. By convention, inspired by the freezing temperature of water at $0°C$, the freezing action is triggered when the control parameter is set to 0, whereas 1 represents the idle situation. Implementing of the freezing control on a Boolean network augments the formulas of the network by adding the control parameter to obtain the expected control behaviour.

A control can be applied to a node. In this case, the control parameters are applied to the formula of the frozen variable. For a formula f_i, adding the control parameters $u_i^0 \in U^0$ and $u_i^1 \in U^1$ to freeze the variable x_i to 0 or 1, respectively, leads to the following specification:

$$x_i = f_i(x_1, \ldots, x_n) \wedge u_i^0 \qquad \text{for freezing to } 0, \tag{2.1}$$

$$x_i = f_i(x_1, \ldots, x_n) \vee \neg u_i^1 \qquad \text{for freezing to } 1. \tag{2.2}$$

A control can be applied an arc. In this case, the control parameters are applied to the occurrence of variables in the functions of other variables. For a formula f_i, adding the control parameters $u_{i,j}^0 \in U^0$ and $u_{i,j}^1 \in U^1$ to freeze the variable x_j in the formula of x_i to 0 or 1, respectively, leads to the following specification:

$$x_i = f_i(x_1, \ldots, x_j \wedge u_{i,j}^0, \ldots, x_n) \qquad \text{for freezing to } 0, \tag{2.3}$$

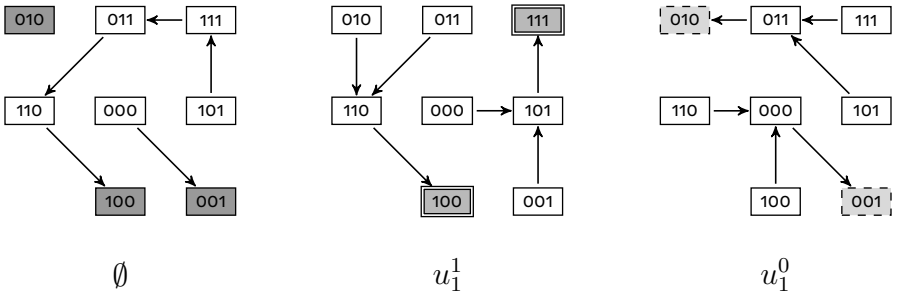
$$x_i = f_i(x_1, \ldots, x_j \vee \neg u_{i,j}^1, \ldots, x_n) \qquad \text{for freezing to } 1. \tag{2.4}$$

Many other types of controls are possible. In this manuscript, we focus on the freezing of nodes and do not consider the control applied on arcs. U^0 and U^1 control parameters can be combined to trigger the freezing to different values (*i.e.*, $x_i = f_i(x_1, \ldots, x_n) \wedge u_i^0 \vee \neg u_i^1$). Subsequently, $U = U^0 \cup U^1$ represents the whole set of freezing control parameters, and $u_i \in U$ represents a generic freezing control parameter (u_i^0 or u_i^1).

The *active control* set of a control input, $\dot{\mu}$, represents the set collecting all the activated controls: $\dot{\mu} = \{u \mid \mu(u) = 0\}$. Note that μ and $\dot{\mu}$ are equivalent descriptions of the control since we can define one from the other. Subsequently, for the sake of simplicity, a control μ will be described using $\dot{\mu}$ to avoid a long description of the control including all inactive control inputs.

In Figure 2.7, we can see an example of the application of a control to the

$$F_{\{u_1^1,u_1^0\}} = \begin{cases} x_1 = \big((x_1 \wedge \neg x_2) \vee (x_1 \wedge \neg x_3) \vee (\neg x_1 \wedge x_2 \wedge x_3) \vee \neg u_1^1\big) \wedge u_1^0 \\ x_2 = (x_1 \wedge x_3) \vee (\neg x_1 \wedge x_2) \\ x_3 = (x_1 \wedge x_3) \vee (\neg x_1 \wedge \neg x_2) \end{cases}$$



Legend: The Boolean network F of Figure 2.1 is completed by the formulas of the freezing controls to produce the Boolean control network $F_{\{u_1^1,u_1^0\}}$. From left to right, the respective controls are: no freeze, x_1 is frozen to 1, x_1 is frozen to 0. The active control parameters are mentioned below each dynamics. The dynamics are synchronous and the self-loops on states are not shown. For the sake of clarity, the set of updated variables m on the transition was omitted. The stable states of each dynamics are coloured in three shades of grey, and their contours are drawn in different styles. Each contour style is associated with a different control input.

Figure 2.7: The synchronous dynamics of a Boolean control network.

variable x_1 on the Boolean network of Figure 2.1. The three depicted dynamics respectively correspond to the following:

1. the absence of control,
2. the freezing of the variable x_1 to 1 ($u_1^1 = 0$),
3. the freezing of the variable x_1 to 0 ($u_1^0 = 0$).

We can see that applying different freezing controls results in different behaviours in the dynamics of the system. These dynamics changes lead to different transitions and different equilibriums.

It is worth noticing that some variables are purposely uncontrolled to play the role of *observers* used for freely reporting the evolution of the states of a system. In biology, *biomarkers* play the role of these observers. An observer is always an uncontrolled variable used to assess the evolution of the system. Therefore, the uncontrolled variables are important for assessing the fate of the dynamical system. The set of controlled variables is denoted C_X, and the set of uncontrolled variables is $\bar{C}_X = X \setminus C_X$. In our example in Figure 2.7, the $\bar{C}_X-$variables are $\{x_2, x_3\}$, and the C_X-variables are $\{x_1\}$

The *profile* of a set of variables A denotes the instantiation of the variables of A in a given state s. In the sequel, the profiles of uncontrolled variables are denoted '$\bar{C}_X-$profiles', and the profiles of controlled variables are denoted 'C_X-profiles'.

3 - Control sequence dynamics

We build on previous studies by characterising control sequences to explore the inference of sequences of control,. Therefore, we need to define a framework that enables us to describe the notion of sequentiality of control in Boolean control networks. In this chapter, we define the control sequence dynamics and related concepts (Section 3.1), present the problem of control sequence discovery (Section 3.2), and discuss its complexity (Section 3.3).

3.1 . Control sequence dynamics

Controlled dynamics extend the Boolean network dynamics by specifying how the system evolves through a sequence of control inputs.

A sequence of controls is formally defined by the function $\mu : \mathbb{N}^+ \to (U \to \mathbb{B})$ indexing control inputs, where $\mu_i, i \geq 1$, is the i-th control input in the sequence and $\mu_{[k]}$ stands for the sequence of size k starting with μ_1 and ending in μ_k: $\mu_{[k]} = (\mu_1, \ldots, \mu_k)$.

Controlled dynamic. Given a Boolean control network F_U, the model of controlled dynamics is defined as a labelled transition system that includes the control inputs as labels $\langle S_X, S_U \times M, \longrightarrow \rangle$, such that as transition is defined by:

$$s \xrightarrow{\mu_i, m} s' \stackrel{\text{def}}{=\!=} s' = (F_{\mu_i})_{\downarrow m}(s) \cup s_{X \setminus m}. \tag{3.1}$$

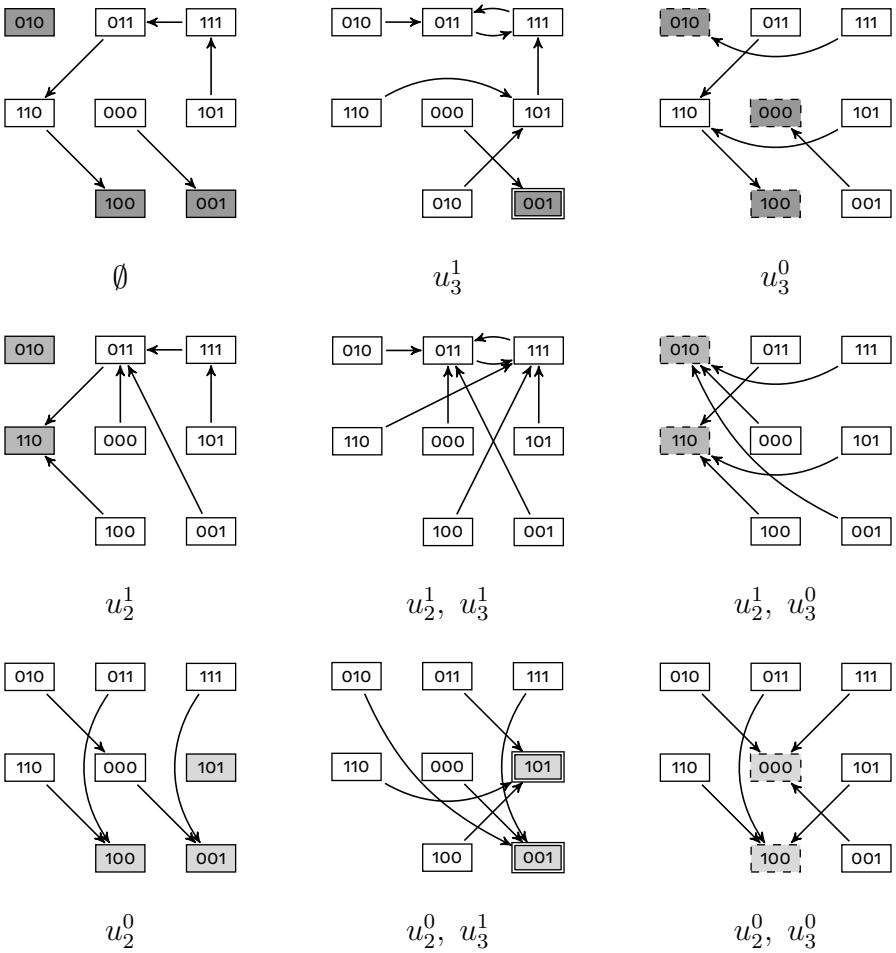
A control sequence $\mu_{[k]}$ leads, therefore, to the following trajectory (path) of size $k+1$ and in which, for each transition, a control is applied:

$$s^1 \xrightarrow{\mu_1, m_1} \ldots s^i \xrightarrow{\mu_i, m_i} s^{i+1} \ldots s^k \xrightarrow{\mu_k, m_k} s^{k+1}.$$

In Equation 3.1 the state s^i denotes the i-th state of the sequence. Each control is labelled by a pair of control and modality. For the sake of clarity, we omit the mode if it is not needed for explanation, meaning we consider the union relation: $\xrightarrow{\mu} = \bigcup_{m \in M} \xrightarrow{\mu, m}$.

State trace. The trace defines the sequence of visited states in the control sequence trajectory: $(s^i)_{1 \leq i \leq k+1}$.

$$F_{\{u_2^1,u_2^0,u_3^1,u_3^0\}} = \begin{cases} x_1 = (x_1 \wedge \neg x_2) \vee (x_1 \wedge \neg x_3) \vee (\neg x_1 \wedge x_2 \wedge x_3) \\ x_2 = ((x_1 \wedge x_3) \vee (\neg x_1 \wedge x_2) \vee \neg u_2^1) \wedge u_2^0 \\ x_3 = ((x_1 \wedge x_3) \vee (\neg x_1 \wedge \neg x_2) \vee \neg u_3^1) \wedge u_3^0 \end{cases}$$



Legend: The Boolean network F of Figure 2.1 is completed by the formulas of the freezing controls to produce the Boolean control network $F_{\{u_2^1,u_2^0,u_3^1,u_3^0\}}$. From left to right, top to bottom, the respective controls are: no freeze, x_3 is frozen to 1, x_3 is frozen to 0, x_2 is frozen to 1, x_2 and x_3 are frozen to 1, x_2 is frozen to 1 and x_3 is frozen to 0, x_2 is frozen to 0, x_2 is frozen to 0 and x_3 is frozen to 1, x_2 and x_3 are frozen to 0. The active control parameters are mentioned below each dynamics. The dynamics are synchronous and the self-loops on states are not shown. The stable states of each dynamics are coloured in three shades of grey, and their contours are drawn in different styles. Each $2-$tuples of contour styles and shades of grey is associated with a different control input. We can notice that the centre graph where x_2 and x_3 is frozen to 1 have a unique cyclic attractor and no stable states.

Figure 3.1: The synchronous dynamics of a Boolean control network.

For example, take the controlled Boolean network described in Figure 3.1 and its different controls. Let the initial state be 010, the sequential application of the sequence of control $\mu[5] = (\{u_2^0\}, \{u_2^0\}, \{u_2^1\}, \{u_2^1\}, \emptyset)$ leads to the following trajectory in the controlled dynamics:

$$010 \xrightarrow{\{u_2^0\}} 000 \xrightarrow{\{u_2^0\}} \mathbf{001} \xrightarrow{\{u_2^1\}} 011 \xrightarrow{\{u_2^1\}} \mathbf{110} \xrightarrow{\emptyset} \mathbf{100}. \tag{3.2}$$

In Equation 3.2, the control inputs are represented by their active control set, with elements indexed by the freeze value, and the stable states traversed by the trajectory are in boldface. The state trace of the trajectory is thus:

$$(010, 000, \mathbf{001}, 011, \mathbf{110}, \mathbf{100})$$

Classes of sequences. Control sequences can be categorised into families based on the evolution of the control between steps. The complexity of the sequential control evolution depends on the rule governing such control application strategies. Indeed, a chosen strategy can impact the inference of a sequence. It may exist a sequence of control for a given strategy that evolves a Boolean network dynamics towards an expected state, when for another strategy such problem cannot be solved. We propose two control application strategies:

- *Total Control Sequence (TCS):* All the controls are triggered during the first phase for all the controlled variables and remain active throughout the sequence. The values to which the variables are frozen may change.

- *Open Control Sequence (OCS):* No constraints regarding control parameters are imposed. A control can be changed or released freely.

The sequence described in the trajectory (3.2) is an OCS since x_2 is controlled, then uncontrolled. The TCS class is mainly used for proofs and algorithmic reasons. This sequence has no realistic biological application of its own. Indeed, in the case where we consider that $\bar{C}_X$-variables correspond to biomarkers, controlling all genes corresponding to C_X-variables is not realistically feasible in vitro and in vivo. The OCS class is the most general class that may represent the action of the drugs on molecular networks, potentially implying the modification and the relaxation of the actions. The following inclusion between these families holds:

$$\text{TCS} \subsetneq \text{OCS}.$$

The control dynamics enable the change of control at any time in the dynamics. This observation leads to the following proposition:

Proposition 3.1. *For any control sequence $\mu_{[k]}$, there exists a total control sequence of the same size $\nu_{[k]} \in$ TCS generating the same state trace under the synchronous mode.*

Proof. Take a control sequence $\mu_{[k]}$ and an initial state. For a transition $s^i \xrightarrow{\mu_i} s^{i+1}, 1 \leq i \leq k$, two cases may occur for the control parameters of the controlled variables, $x_j \in C_X$:

1. If one of the two control parameters u_j^0, u_j^1 is already activated, then the configuration remains the same for ν.

2. If the control parameters are both idle ($u_j^0 = 1, u_j^1 = 1$), then we directly fix the expected final state value by setting the control appropriately, namely: $\nu_i(u_j^0) = 0, \nu_i(u_j^1) = 1$ if $s^{i+1}(x_j) = 0$ and $\nu_i(u_j^0) = 1, \nu_i(u_j^1) = 0$ if $s^{i+1}(x_j) = 1$.

As the update is synchronous, then all the values of the controlled variables x_j lead to the state $s^{i+1}(x_j)$ in a controlled way. For uncontrolled variables, $x_j \in \bar{C}_X$, we have $(f_\nu)_j = (f_\mu)_j$ since no modifications occur, meaning the update is the same.

Since a transition only depends on the previous state that can be obtained by applying of a TCS control input ν_i, we can define a TCS control input for each step, finally leading to a total controlled sequence $\nu_{[k]}$, simulating the trajectory controlled by $\mu_{[k]}$ from an s^1. □

Proposition 3.1 states the observational equivalence between OCS and TCS classes under the synchronous update mode, namely *any OCS control sequence state trace can be reproduced by a TCS sequence.* Thus, under the control dynamics, for each OCS control sequence, a TCS control sequence exists with the same state trace. For example, the OCS sequence of control $\mu[5] = (\{u_2^0\}, \{u_2^0\}, \{u_2^1\}, \{u_2^1\}, \emptyset)$, which generates the trajectory (3.2), has an equivalent TCS sequence of control $\mu'[5] = (\{u_2^0, u_3^0\}, \{u_2^0, u_3^1\}, \{u_2^1, u_3^1\}, \{u_2^1, u_3^0\}, \{u_2^0, u_3^0\})$[1] with the same state trace.

Let the initial state be 010, the control sequence $\mu'[5]$ leads to the following trajectory in the controlled dynamics. The control inputs are represented by their

[1]According to the definition of a TCS, $\mu'[5]$ is also an OCS sequence of control.

active control set, with elements indexed by the freeze value, and the stable states traversed by the trajectory are in boldface.

$$010 \xrightarrow{\{u_2^0,u_3^0\}} \mathbf{000} \xrightarrow{\{u_2^0,u_3^1\}} \mathbf{001} \xrightarrow{\{u_2^1,u_3^1\}} 011 \xrightarrow{\{u_2^1,u_3^0\}} \mathbf{110} \xrightarrow{\{u_2^0,u_3^0\}} \mathbf{100}. \quad (3.3)$$

In control sequence dynamics, allowing a change of control at any time has no concrete application in biology. Knowing the state of the phenotype of a cell at any time to administer a sequential therapy is impractical. On the contrary, waiting for the cell to reach a stable phenotype before changing the administered drug is practicable.

We could have considered other types of attractors; for example, a cyclic attractor for which a subset of components does not vary is instantiated to a desired phenotype. We have chosen to constrain ourselves to stable states due to them being the easiest type of attractors to find. Based on this choice, we define the following control dynamics:

Control Evolution based on Stable–State dynamics. The model of controlled dynamics is said to be *Control Evolution based on Stable–State dynamics* (ConEvs) if the modification of the control inputs instantiation only occurs at a stable state. The change of control modifies the dynamics and releases the stability; hence, the ConEvs dynamics fulfil the following property:

$$\forall s^1 \xrightarrow{\mu_1} \ldots \xrightarrow{\mu_k} s^{k+1} : \mu_i \neq \mu_{i+1} \iff \mathrm{STBL}_{F_{\mu_i}}(s^{i+1}), \quad \text{given that } s^i \xrightarrow{\mu_i} s^{i+1} \xrightarrow{\mu_{i+1}} s^{i+2}, 1 \leq i < k. \quad (3.4)$$

Under the ConEvs dynamics, we impose as a restriction that the property must be validated on a stable state. When no state in the set of target states is stable in the dynamics of at least one of the possible controlled Boolean networks[2], the CoFaSe problem will have no solution.

In ConEvs dynamics, changing the control is the only way to evolve the dynamics since a stable state is reached with the current instance of the Boolean network, which results from applying a control input to the BCN. ConEvs dynamics model either the different mutational steps in which a mutation rewires the network reaching another phenotype, the molecular signature of which is stable,

[2]We also consider the empty control (*i.e.*, the Boolean network not controlled).

or a therapeutic regimen where the drug administering depends on the therapeutic evaluation modelled by a stable state assessment. The trajectory described in (3.2) is ConEvs, in contrast to the trajectory described in (3.3), in which the control is changed in state 011 which is not a stable state of $F_{\{u_2^1,u_3^1\}}$.

Under the ConEvs dynamics, Proposition 3.1 is false. Therefore, some OCS sequences of control do not have a TCS equivalent sequence. Thus, it may exist an OCS sequence of control that evolves a Boolean network dynamics towards a desired state in which, for a TCS control strategy, the same problem is unsolvable. The trajectory described in (3.2) is a prime example of an instance in which no TCS equivalent sequence of control exists. The state trace of the Trajectory (3.2) cross, in this order, the states 001, 011, and 110. To reproduce such trace, two total controls would be needed since the C_X-variables are different in $0\mathbf{01}$, $0\mathbf{11}$ and $1\mathbf{10}$. In Figure 3.1, no controlled Boolean networks have in their dynamics 011 as a stable state. Thus, under the ConEvs dynamics, there is no TCS sequence of control that could reproduce the Trajectory (3.2) state trace.

Contracted control sequence. The *contracted control sequence* retains only one instance of the control input for each sub-sequence with identical control inputs. For ConEvs dynamics, the contracted control sequence can be considered as the sequence making the dynamics evolve from stable states to stable states. Note that under the synchronous update mode and the ConEvs dynamics, the initial control sequence can be easily retrieved by connecting the encountered stable states for each F_{μ_i} using a trajectory controlled by μ_i.The contracted control sequence notation, is therefore, an alternative ConEvs representation of a sequence of control, enabling a clearer visualisation of its control evolution. In the case of Example (3.2), the contracted sequence is thus represented by the active controls $(\{u_2^0\},\{u_2^1\},\emptyset)$.

In the rest of this manuscript, all the results are based on the synchronous dynamics, in which all the variables are updated jointly. This choice was motivated by the fact that, under this update mode, the practical computation time of the reachability problem is reduced. Indeed, in the asynchronous update mode, one needs to tackle the fact that state transition becomes a relation inducing non-determinism, that should not be exhaustively explored to ensure the efficiency of an algorithm.

3.2 . Control sequence discovery

Determining a control sequence that modifies the dynamics to evolve towards an expected state can be stated as a reachability problem:

Let $S_\alpha, S_\omega \subseteq S_X$ be two set of states, can we find a control sequence:
$\mu_{[k]} = (\mu_1, \ldots, \mu_k)$ *such that there exists a path* $s^1 \xrightarrow{\mu_1} \ldots \xrightarrow{\mu_k} s^{k+1}$,
with: $s^1 \in S_\alpha$ *and* $s^{k+1} \in S_\omega$ *?*

We refer to this problem as the *'Controlled Fate in Sequence'* (CoFaSe) problem.

Consider, for example, the Boolean network from Figure 2.7. For this CoFaSe problem, the initial states are $S_\alpha = \{000\}$, whereas the final states are $S_\omega = \{010, 110\}$. x_1 is the sole controllable variable, whereas x_2 and x_3 are observers. Therefore, for this CoFaSe problem, we must find a control sequence reaching any state in which $x_2 = 1$ and $x_3 = 0$ from the initial state 000 by only controlling x_1.

For all three possible controls (*i.e.*, x_1 not freeze, x_1 freeze to 1 and x_1 freeze to 0), no paths connect 000 to 010 or 000 and 110. However, the freeze of x_1 to 1 leads to the state 111 (middle graph). From this state, the freeze of x_1 to 0 (rightmost graph) finally leads to the state 010. This sequence of controls, therefore, solves the above CoFaSe problem and the following trajectory. The control inputs are represented by their active control sets, with elements indexed by the freeze value, and the stable states traversed by the trajectory are in boldface.

$$000 \xrightarrow{\emptyset} \mathbf{001} \xrightarrow{\{u_1^1\}} 101 \xrightarrow{\{u_1^1\}} \mathbf{111} \xrightarrow{\{u_1^0\}} 011 \xrightarrow{\{u_1^0\}} \mathbf{010}. \tag{3.5}$$

Applied to ConEvs dynamics, the CoFaSe problem implies that at least a state appearing in S_ω is stable for F_{μ_k}. *We suggest that all states of S_α should be stable for uncontrolled F.* When this is not the case, we consider the stable states have in their basin of attraction one of the initial states. For example, if we consider for the Boolean network from Figure 2.7 with $S_\alpha = \{000\}$ and $S_\omega = \{010, 110\}$, to change the control, we would need to wait for $F_\emptyset(000)^*$ to reach the stable state 001. In this case, we simply consider that $S_\alpha = \{001\}$. In this case, the TCS contracted control sequence $(\{u_1^1\}, \{u_1^0\})$ resolves CoFaSe under the ConEvs dynamics with the following trajectory, in which all states are stable under the previous control:

$$001 \xrightarrow{\{u_1^1\}}{}^* 111 \xrightarrow{\{u_1^0\}}{}^* 010. \tag{3.6}$$

In another example, let us consider the Boolean network from Figure 3.1 with $S_\alpha = \{010\}$ and $S_\omega = \{100\}$. In this case, the contracted OCS sequence $(\{u_2^0\}, \{u_2^1\}, \emptyset)$ resolves CoFaSe under the ConEvs dynamics with the following

trajectory, in which all states are stable under the previous control:

$$010 \xrightarrow{\{u_2^0\}}{}^{*} 001 \xrightarrow{\{u_2^1\}}{}^{*} 110 \xrightarrow{\emptyset}{}^{*} 100. \tag{3.7}$$

In biological modelling, the outcome of reprogramming can be formulated as a condition on the biomarkers, checking whether the system has reached an expected signature. Note that achieving a given state for *controlled* variables is trivial and consists in merely assigning their expected values by setting the appropriate control inputs. Therefore, the main problem lies in indirectly influencing the state variation of the uncontrolled variables by applying freeze actions on controllable variables.

Minimal control sequences. We now define some properties related to the size of sequences. We first consider the CoFaSe problem, in which the control can be changed at any time:

- A sequence $\mu_{[k]}$ is said to be *minimal* for the CoFaSe problem with respect to F_U, S_α, and S_ω if no control sequences $\nu_{[l]}$ satisfy the CoFaSe problem, such that $l < k$.

We now consider contracted control sequence under the ConEvs dynamics and resolving the CoFaSe $(F_U, S_\alpha, S_\omega)$.

- A contracted control sequence $\mu_{[k]}$ is called *minimal* for the CoFaSe problem with respect to F_U, S_α, and S_ω if *no contracted control sequence* satisfying the CoFaSe problem has a lower number of steps.

To define the following notions on contracted control sequence, we first need to define the parsimony and minimality of a control input in a contracted sequence transition:

- A control input μ is called *parsimonious* for a contracted transition $s \xrightarrow{\mu}{}^{*} s' \wedge STBL_{F_\mu}(s')$ if $\nexists \mu' \in S_U, \mu' \subseteq \mu \wedge s \xrightarrow{\mu'}{}^{*} s' \wedge STBL_{F_{\mu'}}(s')$
- A control input μ is called *minimal* for a contracted transition $s \xrightarrow{\mu}{}^{*} s' \wedge STBL_{F_\mu}(s')$ if $\nexists \mu' \in S_U, |\mu'| < |\mu| \wedge s \xrightarrow{\mu'}{}^{*} s' \wedge STBL_{F_{\mu'}}(s')$. A minimal control input with respect to a given contracted transition is, by definition, also parsimonious.

We now define the parsimony and control minimality of a contracted control sequence:

- A contracted sequence $\mu_{[k]}$ is called *parsimonious* if it yields a trajectory $s^1 \xrightarrow{\mu_1}{}^* s^2 \dots s^k \xrightarrow{\mu_k}{}^* s^{k+1}$ reaching a target state $s^{k+1} \in S_\omega$ from an initial state $s^1 \in S_\alpha$, in which all the controls μ_i, with $1 \leq i \leq k$, are parsimonious controls for the contracted transition $s^i \xrightarrow{\mu_i}{}^* s^{i+1}$.
- A contracted sequence $\mu_{[k]}$ is called *control minimal* if it yields a trajectory $s^1 \xrightarrow{\mu_1}{}^* s^2 \dots s^k \xrightarrow{\mu_k}{}^* s^{k+1}$ reaching a target state $s^{k+1} \in S_\omega$ from an initial state $s^1 \in S_\alpha$, in which all the controls μ_i, with $1 \leq i \leq k$, are minimal controls for the contracted transition $s^i \xrightarrow{\mu_i}{}^* s^{i+1}$. If a sequence is control minimal, it is by definition also parsimonious.

The following sequences are examples of minimal and control minimal contracted control sequences:

- The TCS contracted control sequence $(\{u_1^1\}, \{u_1^0\})$ of Trajectory 3.6 is minimal and control minimal under the ConEvs dynamics.
- The OCS contracted control sequence $(\{u_2^0\}, \{u_1^0\}, \emptyset)$ of Trajectory 3.7 is minimal and control minimal under the ConEvs dynamics. Note that no TCS sequence solving this CoFaSe problem exists.

The following sequences are examples of minimal or control minimal contracted control sequences, but not both:

- Let us consider the Boolean network from Figure 2.7 with $S_\alpha = \{000\}$ and $S_\omega = \{010, 110\}$. The TCS contracted control sequence $(\{u_1^1\}, \emptyset, \{u_1^0\}, \{u_1^1\}, \{u_1^0\})$ that resolves CoFaSe under the ConEvs dynamics with the following trajectory is a control minimal contracted control sequence but is not minimal:

$$001 \xrightarrow{\{u_1^1\}}{}^* 111 \xrightarrow{\emptyset}{}^* 100 \xrightarrow{\{u_1^0\}}{}^* 001 \xrightarrow{\{u_1^1\}}{}^* 111 \xrightarrow{\{u_1^0\}}{}^* 010. \tag{3.8}$$

- Let us consider the Boolean network from Figure 3.1 with $S_\alpha = \{010\}$ and $S_\omega = \{100\}$. The OCS contracted control sequence $(\{u_2^0\}, \{u_2^1\}, \{u_2^0, u_3^0\})$ that resolves CoFaSe under the ConEvs dynamics with the following trajectory is a minimal contracted control sequence but is not control minimal:

$$010 \xrightarrow{\{u_2^0\}}{}^* 001 \xrightarrow{\{u_2^1\}}{}^* 110 \xrightarrow{\{u_2^0, u_3^0\}}{}^* 100. \tag{3.9}$$

Note that a minimal and control minimal sequence for a given CoFaSe problem may not be the sequence with the minimal total number of perturbations (*i.e.*, number of freeze or unfreezing of variables). Indeed, there may be a control sequence with a minimal total number of perturbations that is not minimal in size.

3.3 . Complexity of CoFaSe

In this section, we show that the inference of a control sequence satisfying CoFaSe is PSPACE-hard. Since the freezing to 0 and to 1 cannot be triggered simultaneously for a single variable, the cardinality of possible controlled transitions from a state is $3^{|X|} \cdot |M|$. Finding a single parsimonious control without considering a set of initial states is NP-complete [7]. Thus, the CoFaSe problem is even less tractable than finding single controls (assuming that PSPACE $\neq$ NP). Therefore, finding the control sequence by exhaustively exploring possible control spaces is not tractable.

To prove that CoFaSe inference is a PSPACE-hard problem, we lean on the fact that the problem of reachability in Boolean networks working in synchronous mode can actually be formalised as a CoFaSe problem. Indeed, reachability in a Boolean network is precisely the CoFaSe problem for a Boolean control network without controlled variables. Lemma 3.1 shows that this reduction is not merely an artefact. Indeed, We can construct a network with a non-empty set of control variables and reduce the CoFaSe problem for this network to a reachability problem for a standard Boolean network.

Lemma 3.1. *Deciding whether a control sequence exists for the CoFaSe problem in the synchronous mode is at least as hard as reachability in (uncontrolled) Boolean networks in synchronous mode.*

Proof. Take an n-variable Boolean network F and construct a Boolean control network F' by adding to F the single control variable x_0 and defining the update functions f'_i of F' in terms of the update functions f_i of F in the following way:

$$\begin{aligned} f'_i &= f_i \wedge x_0, 1 \leq i \leq n, \\ f'_0 &= 0, \end{aligned}$$

where f'_0 is the update function for x_0.

Consider the controls $\mu_1 = d_0^1$ and $\mu_0 = d_0^0$ controlling x_0 to 1 and 0 respectively. The previous two properties ensure that the state graph of F'_{μ_1} is that of F, with $x_0 = 1$ added to each state, and that the state graph F'_{μ_0} only

contains transitions to the state $\mathbf{0}$, which is the state in which all variables are 0.

Let X be the set of variables of F. The set of variables of F' is thus $X' = X \cup \{x_0\}$. Fix a set of starting states $S_\alpha \subseteq S_{X'}$ and a set of target states $S_\omega \subseteq S_{X'} \setminus (S_\alpha \cup \{\mathbf{0}\})$, such that the states in both sets satisfy $x_0 = 1$. The CoFaSe problem for the tuple (F', S_α, S_ω) has a solution if and only if the states in $S_{\omega\downarrow X}$ are reachable from $S_{\alpha\downarrow X}$ in F. Indeed, by construction of F' and since $\mathbf{0} \notin S_\omega$, the control sequence for this instance of CoFaSe may only be the singleton control sequence consisting of μ_1, and it must ensure the reachability of S_ω from S_α in F_{μ_1}, whose state graph is trivially isomorphic to that of F.

In conclusion, an oracle for CoFaSe would allow to solve reachability in Boolean networks working in the synchronous mode with, at most, polynomial overhead, which proves the lemma statement. □

It is known that the complexity of the reachability in Boolean networks working in synchronous mode is PSPACE-complete [44, 21].[3] Below, we provide a sketch of an alternative proof of this complexity based on a reduction from Deterministic Linear Bounded Automaton (Lemma 3.2). In Lemma 3.3 we prove that the reachability of Boolean networks with the synchronous mode is PSPACE. Finally, these two lemmas result in the Theorem 3.1, which state the PSPACE-completeness of the reachability problem for Boolean networks with the synchronous update mode.

Lemma 3.2. *Given a Boolean network F with the variables X, a set of starting states $S_\alpha \subseteq S_X$, and the set of target states $S_\omega \subseteq S_X \setminus S_\alpha$, it is PSPACE-hard to decide whether F can reach any of the states in S_ω from a state in S_α in synchronous mode.*

Proof. The proof idea is to polynomial-time reduce the acceptance problem of a *Deterministic Linear Bounded Automaton* (a DLBA) to reachability for Boolean networks working in synchronous mode.

An LBA is a Turing machine that is only allowed to use, at most, $f(n)$ contiguous tape cells, in which n is the size of the input and f is a linear function. Deciding whether a DLBA accepts a given input string is a PSPACE-complete problem (e.g., [23]).

Take a DLBA M and construct the Boolean network F simulating M in the following way. Define the Boolean variables $A_{i,j}$ and $Q_{i,k}$, in which i indexes

[3] In [21], Dennunzio et al. prove that the reachability problem in reaction systems is PSPACE-complete. Since reaction systems form a subclass of Boolean networks, the PSPACE-hardness result is a lower complexity bound for the reachability problem in Boolean networks.

the tape cells of M, j indexes the symbols in the tape alphabet of M, and k indexes the states of M. The situation in which the i-th tape cell contains the j-th symbol is represented by setting $A_{i,j}$ to 1. The situation in which M is in the k-th state and the head is on the i-th tape cell is represented by setting $Q_{i,k}$ to 1. F operates by stepwise simulating the evolution of M: rewriting the j_1-th symbol to the j_2-th symbol in the i-th tape cell is done by setting A_{i,j_1} to 0 and A_{i,j_2} to 1, while moving the head from cell i_1 to i_2 and changing the state from k_1 to k_2 is simulated by setting Q_{i_1,k_1} to 0 and Q_{i_2,k_2} to 1. The synchronous dynamics of F, therefore, faithfully simulate M, because M is deterministic.

For any input word w, the DLBA M reaches a configuration in the set of accepting configurations C_A if and only F can reach the encoding of one of the configurations C_A from the encoding of the initial configuration of M. The statement of the lemma follows from the facts that the procedure of constructing F from M is polynomial and that acceptance for DLBA is PSPACE-complete. □

Lemma 3.3. *Given a Boolean network F with the variables X, a set of starting states $S_\alpha \subseteq S_X$, and the set of target states $S_\omega \subseteq S_X \setminus S_\alpha$, it is in PSPACE to decide whether F can reach any of the states in S_ω from one of the states in S_α in synchronous mode.*

Proof. The proof idea is to construct a DLBA M that accepts the input if and only if the Boolean network F can reach a state in S_ω from a state in S_α. The initial configuration of M consists of the following three segments:

1. the list of binary vectors representing the states in S_α, each vector written in two copies;
2. the list of binary vectors representing the states in S_ω, each vector written in one copy;
3. an $|X|$-bit binary counter initialised to 0, where $|X|$ is the number of binary variables of F.

In the remainder of the proof, we implicitly assume that the states of F are represented as binary words. A state $(x_1, x_2, ..., x_n)$ is thus represented by the word $x_1x_2 \dots x_n$.

Consider a state $s \in S_\alpha$. The initial configuration of M contains a substring ss. M starts by simulating the transitions of F on one copy of s and

replacing the other copy by the new state $s' = F(s)$, thereby yielding the new substring ss'. The subsequent operation of M is divided into macrosteps, during which it carries out the following actions:

1. calculates the new state for each pair of states in Segment (1);
2. compares each new state with the states written in Segment (2); if one of these comparisons is successful, M accepts, otherwise it continues to the following substep;
3. checks if all the bits of the binary counter in Segment (3) are 1; if yes, reject, otherwise, commence the next macrostep.

Intuitively, M simulates the deterministic synchronous dynamics of F on every state in Segment (1), accepts if it sees a target state from S_ω, or rejects after $2^{|X|}$ steps. Counting to $2^{|X|} = |S_X|$ ensures that the entire state graph of F reachable from S_α is visited. Therefore, M accepts if and only if F can reach at least one state in S_ω from at least one state in S_α. Constructing M from the triple (F, S_α, S_ω) is a polynomial-time procedure, meaning that an oracle for DLBA acceptance would allow deciding reachability for Boolean networks working in the synchronous mode with polynomial overhead. This point proves the statement of the lemma. □

Theorem 3.1 below is derived directly from Lemmas 3.2 and 3.3.

Theorem 3.1. *Given a Boolean network F with the variables X, a set of starting states $S_\alpha \subseteq S_X$, and the set of target states $S_\omega \subseteq S_X \setminus S_\alpha$, it is PSPACE-complete to decide whether F can reach any of the states in S_ω from one of the states in S_α.*

Theorem 3.1 combined with Lemma 3.1, implies the complexity of CoFaSe is at least PSPACE-hard. This point is stated in Theorem 3.2. Whether solving CoFaSe is in PSPACE remains an open question.

Theorem 3.2. *Deciding the existence of a control sequence for the CoFaSe problem in the synchronous mode is PSPACE-hard.*

4 - State of the art in control inference

Algorithmic methods enabling the identification of control strategies of Boolean networks in order to identify therapeutic targets have been the subject of a number of publications in recent years. The controllability of biological systems has mostly been done in one-step. Sequential controllability analysis, in contrast, is still in its infancy. and has mainly been studied by Mandon et al. in [35, 37, 36, 34]. We therefore explain their approach in detail and compare it with our works.

In this chapter, we first describe different approaches of one-step programming in Section 4.1. In Section 4.2, we then describe the sequential programming approach proposed by Mandon et al. Finally, we will conclude in Section 4.3 by explaining the principal differences between the sequential paradigm proposed by Mandon et al. and ours.

4.1 . One-step reprogramming

We briefly recall the existing approaches to one-step reprogramming, namely: simulation, max-SAT ATPG,attractors and Hamming distance, stable motifs, Gröbner basis, and prime implicants. These approaches differ according to their optimisation objectives and control application.

4.1.1 . Simulation

To the best of our knowledge, one-step reprogramming for Boolean networks was pioneered by Layek et al. in [30]. The authors present an approach for designing cancer therapies by assimilating cancerous perturbations with stuck-at-fault and bridging faults by analogy with the errors of electronic circuits. Relevant gene regulation pathway information is first used to produce an acyclic Boolean network. The resulting network has a set of input and output variables. The Boolean network is then transformed into a digital circuit from which an enumeration and classification of possible faults is realised.

Stuck-at-fault corresponds to the setting of a network variable to a particular value (0 or 1). Bridging faults correspond to the removal or incorporation of new interactions between system variables. A simulation-based method for implementing and identifying stuck-at-fault failures is proposed. The authors first model, from biological knowledge, the effect of existing n drugs. The authors then exhaustively simulate from a given initial state the 2^n possible drug binary vectors. If

a particular drug is applied, it is assigned the value 1; otherwise, it is assigned the value 0. A drugs vector is considered effective if, for single failures or combinations of failures, its application to the system results in a healthy output state. The method proposed in this article is therefore based on a 'brute force' method of the exhaustive simulation of all possible single failures for acyclic Boolean networks.

4.1.2 . Max-SAT ATPG

In [32], Lin et al. propose an improvement on the method by Layek et al. The authors present an efficient and extensible SAT-based ATPG[1] methodology for cancer therapy and introduce the notion of optimality of therapies. The digital circuit presented in [30], in addition to the testing conditions, is converted into a conjunctive normal form (CNF). The conjunctive normal form is then augmented with the desired output and solved using a weighted partial Max-SAT solver. Each gene of the network is associated with an CNF formula. This formula is evaluated at 1 if and only if the variables representing its inputs and outputs take on values consistent with the gene truth table. An α failure is added to this formula as an input variable. The simulation of the activation of the failure consists in adding a clause α to the CNF, whereas the simulation of the non-activation of the failure consists of adding a clause $\neg\alpha$ to the CNF. To select the drugs that guarantee the best output from the network, weights are assigned to the clauses representing exit variables in the conjunctive normal form so that the healthy state is associated with the greatest weight, and the failed states the lowest with weight. In order to prioritise treatments that minimise the number and cost of drugs, positive weights proportional to the cost of a drug are assigned to failures corresponding to the action of the drug.

4.1.3 . Attractors and Hamming distance

In [42, 43], Paul et al. propose an approach for finding single-step control, enabling the Boolean network to reach one of the desired attractors from a given initial state. The paper considers various types of control actions but focuses on temporary simultaneous perturbations. The authors aim is to make the control as least invasive as possible to the system as possible. This aim results in the search for the most parsimonious control. Simultaneous temporary perturbations were all applied for just a single time step at the same time. The perturbations induce a change in the value of some variables, but these variables can later be

[1]Automatic Test Pattern Generation and Automatic Test Pattern Generator is an electronic design automation method used to find an input that, when applied to a digital circuit, enables to automatically distinguish between correct circuit behaviour and faulty circuit behaviour caused by defects.

updated according to their original Boolean function. Thus, only the dynamics of the Boolean network without control needs to be considered.

The problem of finding a minimal control driving the system from an initial state to a desired attractor is PSPACE-hard. Thus, a simple global approach performing computations on the entire network will not scale well for large networks. Therefore, they propose a decomposition-based solution to this problem, which can be significantly quicker than existing approaches on large networks. This approach takes advantage of existing algorithms for computing the basin of attraction of an attractor.

The central assumption of Paul et al. is that the computed basins of attraction are smaller than the size of the transition system. This factor reduces the state space that needs to be considered and thus improves efficiency. A minimal control is determined by finding the minimal Hamming distance between the initial state and the state of a target attractor. The control can then be deduced from the substitutions of variable values required to jump from the initial state to the state of the target attractor. The algorithm presented in [43] performs efficiently for networks with a low number of small strongly connected components in the interaction graph as, in this case, the computed basins of attraction will also be small.

4.1.4 . Stable motifs

In this article [61], Zañudo et al. propose a network control approach that combines the structural and functional information of a Boolean network to identify control targets. The method builds on the concept of stable motifs and their relation to finding attractors. By connecting stable motifs with other stable motifs, the presented algorithm identifies targets whose manipulation ensures the convergence of the system to an attractor of interest form the original network dynamics. Stable motifs are defined as interaction subgraphs composed of minimal strongly connected components in which the states of the motif variables form a partial fixed point that, once reached, will not be changed by the dynamics. Stable motif control interventions are guaranteed to drive the network from all possible sets of initial states to the target attractor state. This outcome can be explained by the fact that controlling all cycles of a Boolean network results in a network with only one attractor. Furthermore, the control only needs to be applied transiently for the network to reach and stay in the desired attractors of the original network.

4.1.5 . Gröbner basis

The authors of [39], Murrugarra et al., propose an approach that takes advantage of the rich algorithmic theory of computer algebra to infer potential intervention targets in synchronously updated Boolean networks. The paper considers two types of control actions: the deletion of arcs and the deletion (or constant expression) of nodes. The proposed methods are based on rewriting the Boolean network, its control actions, and the desired property as a system of polynomial equations. Three control actions are considered in the paper: the generation of new steady states representing a desirable cell fate,; the removal of existing steady states representing undesirable cell fates; and the blocking of regions in the state space in which particular values of variables trigger an undesirable pathway or are the signature of an abnormal cell. Based on these points, the authors obtain a system of polynomial equations (or a single equation) that needs to be solved to find the appropriate controls. The resolution of this system of equations is based on the calculation of the Gröbner basis. The controls inferred by using this approach do not guarantee the global reachability of a desired stable state because the inferred control modified the state space by augmenting the size of a desired stable state basin of attraction or removing undesired attractors.

4.1.6 . Prime implicants

Biane et al. propose in [9, 7, 8] an approach for inferring the minimal sets of actions, enabling the reprogramming of Boolean networks updated synchronously or asynchronously. The authors introduce the formalism of controlled Boolean networks via arc freezing and variable freezing. The control formalism presented in Section 2.3 is drawn from the work of Biane et al. who present two reprogramming modes, possibility and necessity, in which, respectively:

- At least a stable state of the controlled Boolean network validates the desired property.
- All stable states of the controlled Boolean network validate the desired property.

Biane et al. prove that the computation of the parsimonious control strategy, in which the network needs to have stable states validating a property, is a problem of abductive inference in propositional logic. Using well-known methods for computing the prime implicants of Boolean functions, the authors developed algorithms computing all parsimonious control strategies.

The first algorithm recursively calculates all parsimonious control strategies. The algorithm is based on integer linear programming and rewrites the Boolean

network, its control actions, and the desired property as a system of linear equations. This method also makes it possible to assign costs to reprogrammation and thus compute control strategies minimizing this cost. This algorithm is also used as a solver in one of the algorithms solving CoFaSe under the ConEvs dynamics that we present later.

The second algorithm based on the binary decision diagram calculates all parsimonious control strategies in one-step. This algorithm calculates the prime implicants from the formula specifying the stability and property conditions represented by a reduced ordered binary decision diagram. This algorithm obtains better performance than the first algorithm in the case of the necessity reprogramming mode.

4.2 . Sequential reprogramming

In [35, 37, 36], Mandon et al. studied asynchronously updated Boolean networks and, more particularly, their interaction graph. The authors expected to find all control sequences reaching a desired property from a set of initial states and having a number of disturbances inferior to a defined limit. The authors considered the total number of perturbations realised throughout the sequence. A minimal sequence of control is considered to be a sequence with a minimal number of perturbations (*i.e.*, a minimal number of controlled variables). The authors goal was to find sequential controls with fewer perturbations to bring new reprogramming solutions to biological networks for which one-step reprogramming strategies are already known.

To modify the dynamics of a Boolean network, Mandon et al. perturb either one of its functions or its current state. A function perturbation is considered to be a permanent perturbation as it induces a permanent state change of nodes to a desired profile of 1 or 0. Such perturbation is equivalent to node freezing. A state perturbation is considered to be a temporary perturbation as it induces a state change of nodes, but these nodes can later be updated according to their original Boolean functions. A state perturbation can also be viewed as a jump in the dynamics of the studied Boolean network.

Given a set of initial states and a set of target states, Mandon et al. classify successful reprogramming strategies according to different 'degrees of success':

- If the target is reachable in principle with the control strategy but might never be reached then the strategy is called existential.

- If the target is always reached, then the strategy is called inevitable.

Paper [35] shows that the strongly connected components of the interaction graph considerably influence the dynamics of the Boolean network. The authors found that strongly connected components that contain loops are more important. Paper [34] presents an algorithm that perturbs the strongly connected components in a given order, using sequentiality to reduce the size of the perturbations. However, the proposed algorithm does not always return minimal solutions and works only on networks with small strongly connected components.

Papers [36, 37] propose a second algorithm that relies on a transition graph with k superposed layers corresponding to the number of k possible perturbations given by the user. This second algorithm, which enables the network to be perturbed in any state, returns a complete list of solutions for temporary and permanent perturbations but only works on small networks due to its slowness.

In the framework of the papers [35], sequential reprogramming enables the network to be perturbed in any state. The authors claim this process requires complete observability of the system, which is very difficult to obtain experimentally in *vivo* and in *vitro*. To make the sequential reprogramming practical, the authors designed a sequential reprogramming strategy that uses attractors [42, 43] as intermediate steps. This control update dynamics resembles our ConEvs dynamics but, unlike ConEvs, enable a change of control on cyclic attractors. The authors propose a third algorithm [36, 37] using this new update control dynamics and inspired by their second algorithm, enabling only temporary perturbations. This algorithm is quicker than the other proposed methods in some cases and returns shorter perturbation sequences. However, the resulting sequences are fewer and generally have a larger number of perturbations than those yielded by the second algorithm.

4.3 . Conclusion

The algorithmic methods developed in the literature to calculate control strategies for Boolean networks differ in the objectives of the reprogramming, the modelled perturbation, the dynamics and topology of the studied networks, and the minimality of the solutions. For example, the one-step controllability methods based on simulation in [30, 32] and attractors and Hamming distance in [42, 43] seek to infer controls that drive the system from *a stable state* to an attractor validating a desired property. In contrast, the one-step controlabillity methods based on stable motifs [61], the Gröbner basis in [39], and the prime implicants in [9, 7, 8] seek to infer controls that drive the system from *all stable states* to a set of attractors validating a desired property. In the literature, the main approaches have some similarities but also some differences in term of goal and framework.

For example, determining a control driving the system from an initial state to a desired attractor and determining a control resulting in a network with a given set of property on its fixed point attractors are two problems with differing complexities. The first is PSPACE-hard, whereas the second is NP-hard. Such differences mean that comparing these approaches can be a difficult exercise.

The sequential reprogramming framework we propose in this thesis differs from that introduced in [35, 37, 36, 34]. We seek to infer a minimal contracted control sequence in size. In contrast, Mandon et al. 's main objective was to minimise the total number of perturbations, particularly regarding known one-step controls. These fundamental differences in the objectives of the reprogramming and the definition of minimality form real variations in the way sequential control is defined and treated. Concretely, in all examples in which the algorithms by Mandon et al. found a sequential control smaller than an existing one-step control, our algorithms would return only a one-step control.

Another difference concerns the nature of perturbations: the third algorithm of Mandon et al. presented in [36, 37] only addresses with temporary perturbations. In this case, the dynamics graph of the controlled Boolean network does not change with the control. This outcome differs from our paradigm of Boolean controlled network, in which each control can result in a different dynamics graph. Thus, despite similarities such as the importance of strongly connected components, which we explore later, our method deals with a slightly different problem.

5 - Dynamical sequence analysis

Finding a minimal control sequence by exhaustively exploring all possible control sequences is not tractable. Indeed, since controllable variables can alternate between being controlled to 1, 0 or uncontrolled indefinitely, the number of possible control sequences is infinite. Therefore, we need to design a method capable of inferring an appropriate control sequence without performing an exhaustive exploration of all the sequences.

In the trajectory of a minimal sequence $\mu_{[k]}$, the set of intermediary states s^i, $1 < i \leq k$ can be viewed as intermediary properties that must be reached before reaching a target state. By seeking for factors that limit the set of possible intermediary states, we should significantly reduce the search space in practice.

In order to find such factors, we study the dynamical properties of the CoFaSe problem generated by the opposition between $\bar{C}_X$-variables and C_X-variables. Proposition 3.1 states that when not in ConEvs dynamics, the dynamics of controllable variables can be ignored as they can be reproduced by a TCS sequence. This statement offers insights into the critical role of the uncontrolled variables in the resolution of the CoFaSe problem. As the variables in C_X are fully controlled by the TCS sequence, their natural evolution is essentially discarded. Thus, visiting an already encountered $\bar{C}_X$-profile with a different C_X-profile is irrelevant for solving a CoFaSe problem. Therefore, the dynamical properties that imply the necessity of a sequence of control should emerge from the update functions of $\bar{C}_X$-variables. The same observation is true for solving CoFaSe under the ConEvs dynamics.

In this chapter, we explore the dynamical properties of the CoFaSe problem emanating from the existence of a set of C_X-variables. In the first section, we study the partitioning of the $\bar{C}_X$-variables and their properties. In the second section, we define a bound on the size of sequence in the CoFaSe problem. In the final section, we present bounds for the ConEvs dynamics.

5.1 . Partitioning of the $\bar{C}_X$-variables

This technical section defines the notions employed to solve the proofs of the upper bounds on sequence length (Sections 5.2 and 5.3).

5.1.1 . Partitioned dynamics.

We focused on the partitioning of the states of the Boolean networks with respect to their $\bar{C}_X$-profiles. Starting from a model of dynamics, we defined the quotient graph called *partitioned dynamics*, representing the transitions over the partition. The following equivalence relation is used for its definition:

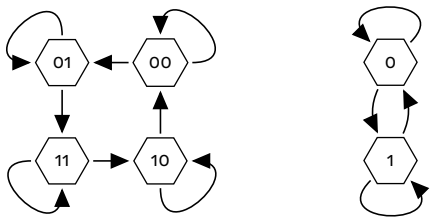
$$\forall s, s' \in S_X : s \sim s' \Leftrightarrow s_{\bar{C}_X} = s'_{\bar{C}_X}. \tag{5.1}$$

All the states with the same $\bar{C}_X$-profile belong to the same partition. Formally, the partitioned dynamics of the Boolean network F_U is, thus, a labelled transition system $\langle S_{\bar{C}_X}, U, \twoheadrightarrow \rangle$. $[s]$ denotes the equivalence class of the state s according to the equivalence relation (5.1). The transition is defined as follows:

$$[s] \twoheadrightarrow [s'] \stackrel{\text{def}}{=\!=} s, s' \in S_X, \exists \mu \in S_U : s \xrightarrow{\mu} s'. \tag{5.2}$$

The resulting partitioned dynamics models the intrinsic dynamical interactions between the signature variations of the biomarkers and provides an overview of the possible sequence-controlled evolution of the network.

Figure 5.1 displays the partitioned synchronous dynamics of the example Boolean networks from Figure 2.7 and Figure 3.1.



Partitioned dynamics of $F_{\{u_1^1, u_1^0\}}$

Partitioned dynamics of $F_{\{u_2^1, u_2^0, u_3^1, u_3^0\}}$

Legend: On the left is the synchronous partitioned dynamics of the Boolean network $F_{\{u_1^1, u_1^0\}}$ of Figure 2.7 where $\bar{C}_X = \{x_2, x_3\}$ and on the right is the synchronous partitioned dynamics of the Boolean control network $F_{\{u_2^1, u_2^0, u_3^1, u_3^0\}}$ of Figure 3.1 where $\bar{C}_X = \{x_1\}$. Equivalence classes are represented by polygons containing the $\bar{C}_X$-profile of the states belonging to them.

Figure 5.1: Synchronous partitioned dynamics of two controlled Boolean networks.

5.1.2 . Properties of equivalence classes.

In this subsection, we define the properties of equivalence classes and the properties of states belonging to equivalence classes.

Target equivalence class. Take s as a state; if $\exists \mu \in S_U, \exists s' \in S_X : s \xrightarrow{\mu} s'$, then $[s']$ is defined as a *target equivalence class* of s. Propositions 5.1 and 5.2 provide observational properties of target equivalence classes and their states in synchronous evolution.

Proposition 5.1. *In the synchronous mode, regardless of the control, each transition from a state leads to states belonging to a unique target equivalence class.*

Proof. Assume an initial state s with s^1 and s^2 two states derived from the one step synchronous evolution of, respectively, F_{μ_1} and F_{μ_2}. Assume also that the following is true:

$$\exists \mu_1, \mu_2 \in S_U : s \xrightarrow{\mu_1} s^1 \wedge s \xrightarrow{\mu_2} s^2 \wedge s^1_{\bar{C}_X} \neq s^2_{\bar{C}_X}$$

By definition $\bar{C}_X$-variables are not controllable. In synchronous mode, the dynamics is determined by anterior states (*i.e.*, applying different controls cannot change the values of the $\bar{C}_X$-variables, because one step is not enough to propagate the updates). Thus, the states s^1 and s^2 cannot have different equivalence classes by means of controls. This point contradicts the above equation and substantiates the statement of this proposition. □

Proposition 5.2. *In the synchronous mode, a state can reach any state of its target equivalence class in a single step under an appropriate control.*

Proof. The proposition follows from the fact that, in the synchronous mode, all C_X-profiles are reachable by a total control in a single step □

Impermanent and enduring states. For a better understanding of the evolution of states with respect to equivalence classes, we distinguish *impermanent states* which cannot be stabilised whatever the applied control, from *enduring states* that can be stabilised by an appropriate control. This point is formally expressed as follows:

$$\forall \mu \in S_U : \neg STBL_{F_\mu}(s) \qquad \text{impermanent state,} \tag{5.3}$$

$$\exists \mu \in S_U : STBL_{F_\mu}(s) \qquad \text{enduring state.} \tag{5.4}$$

Any state s must either be impermanent or enduring: the definitions in Equations (5.3) and (5.4) are logical opposites. We can further derive $\bar{C}_X$-related properties from these definitions.

- If s is impermanent, s cannot be a stable state for any control. Therefore, we can deduce the following equation $\forall \mu : F_\mu(s) \neq s$. We know that this point true even for the total control μ^T, which fixes the C_X-variables to their values in s (*i.e.*, $(F_\mu(s))_{C_X} = s_{C_X}$). According to Proposition 5.1, no matter the control, $(F_\mu(s))_{\bar{C}_X}$ will always have the same profile. Thus, s must be a state in which the $\bar{C}_X$-variables must evolve at the next dynamics step, $\forall \mu : (F_\mu(s))_{\bar{C}_X} \neq s_{\bar{C}_X}$.
- If s is enduring, then there must exist a control μ where $F_\mu(s) = s$. According to Proposition 5.1, no matter the control, $(F_\mu(s))_{\bar{C}_X}$ will always have the same profile. Therefore, $\forall \mu : (F_\mu(s))_{\bar{C}_X} = s_{\bar{C}_X}$.

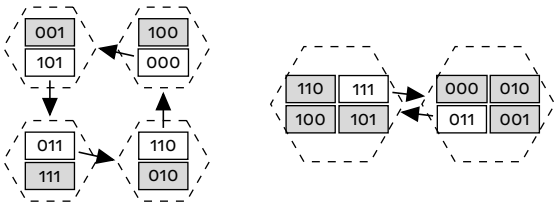
Therefore, an arc between two equivalence classes $[s]$ and $[s']$ in the partitioned dynamics implies the existence of a state in $[s]$ whose target equivalence class is $[s']$. If $[s] = [s']$, s must be an impermanent state. In the case where $[s] \neq [s']$, s must be an enduring state. These dynamical properties are formally defined as follows:

$$\begin{aligned} &s \text{ is impermanent} \quad \text{if: } \forall \mu \in S_U : s \xrightarrow{\mu} s' \implies s_{\bar{C}_X} \neq s'_{\bar{C}_X}, \\ &s \text{ is enduring} \quad \text{if: } \forall \mu \in S_U : s \xrightarrow{\mu} s' \implies s_{\bar{C}_X} = s'_{\bar{C}_X}. \end{aligned}$$

Figure 5.2 shows the partitioned dynamics of the Boolean networks from Figures 2.7 and 3.1, together with the impermanent and enduring states, as well as their target equivalence classes.

5.2 . Bounds on sequence size

Proposition 5.2 and the definition of an impermanent state provide insights into the resolution of the CoFaSe problem. The dynamical properties related to equivalence classes enable us to define an upper bound on the length of minimal control sequences. A minimal sequence solving the problem should jump from an impermanent state to another until it reaches the equivalence class of a target state. A trajectory induced by a minimal control sequence should not pass through intermediary enduring states. Such an action would be redundant as all states of an impermanent target equivalence class of a state s are always reachable with a



Enduring and impermanent states of $F_{\{u_1^1,u_1^0\}}$

Enduring and impermanent states of $F_{\{u_2^1,u_2^0,u_3^1,u_3^0\}}$

Legend: On the left are the enduring and impermanent states of the Boolean network $F_{\{u_1^1,u_1^0\}}$ of Figure 2.7 where $\bar{C}_X = \{x_2, x_3\}$ and on the right are the enduring and impermanent states of the Boolean control network $F_{\{u_2^1,u_2^0,u_3^1,u_3^0\}}$ of Figure 3.1 where $\bar{C}_X = \{x_1\}$. The enduring states of each equivalence class of the partitioned dynamics are coloured in grey and the impermanent states, in white, are connected to their target equivalence classes represented by a dotted circle.

Figure 5.2: Synchronous enduring and impermanent states and the partitioned dynamics of two Boolean networks.

control from s.

Theorem 5.1 defines an upper bound on the size of minimal sequences that only depends on the number of equivalence classes and, thus, of uncontrolled variables. This theorem reveals the critical role of uncontrolled variables in determining the control sequence. Indeed, the upper bound given by Theorem 5.1 ($2^{|\bar{C}_X|}$) depends on the number of $\bar{C}_X$-variables.

Theorem 5.1. *The size of the minimal control sequence $\mu_{[k]}$ solving a given CoFaSe problem is bounded by $2^{|\bar{C}_X|}$ for the synchronous mode: $\left|\mu_{[k]}\right| \leq 2^{|\bar{C}_X|}$.*

Proof. Assume that $\mu_{[k]}$, with $k > 2^{|\bar{C}_X|}$, is a minimal sequence solving the CoFaSe problem $\langle F_\mu, S_\alpha, S_\omega \rangle$. The control sequence $\mu_{[k]}$ yields the following sequence of states: $s^1 \xrightarrow{\mu_1} s^2 \dots s^k \xrightarrow{\mu_k} s^{k+1}$, with $s^1 \in S_\alpha$ and $s^{k+1} \in S_\omega$.

From Proposition (5.2), we know that all states of a target equivalence class are reachable in one step under the appropriate control. Since only the update of impermanent states evolves the uncontrolled variables, it is only possible to reach the next desired impermanent state or property from another impermanent state.Then, the main steps for solving CoFaSe consist in finding the minimal path, from a state of S_α to a state of S_ω, by traversing solely impermanent through impermanent states of different equivalence classes.

Therefore, a minimal control sequence does not pass through two states with identical equivalence class. Since $\mu_{[k]}$ is minimal, the states from s^2 to s^k are impermanent states and the equivalence classes from $[s^2]$ to $[s^k]$ are different target equivalence classes of their former state. The sequence then spans $k-1 \geq 2^{|\bar{C}_X|}$ different equivalence classes. Since only $2^{|\bar{C}_X|}$ $\bar{C}_X$-profiles exist, for $2 \leq i < j \leq k+1$, the sequence must have a state s^i with the same $\bar{C}_X$-profile as s^j. Note that $2 \leq i$ because the first state s^1 may not be an impermanent state. Thus, from the state s^{i-1} having for target equivalence class $[s^i] = [s^j]$, it is possible to reach s^j and to yield the following sequence $s^1 \xrightarrow{\mu_1} \dots s^{i-1} \xrightarrow{\mu'_{i-1}} s^j \xrightarrow{\mu_j} \dots s^{k+1}$. Hence, the sequence $\mu_{[k]}$ is not minimal, thus contradicting the original assumption and proving the statement of the theorem. □

For example, consider the Boolean network from Figure 2.7 with $S_\alpha = \{000\}$ and $S_\omega = \{010, 110\}$. By following the minimal path between the equivalence classes $[000]$ and $[110]$ or [010] in the right graph of Figure 5.2, we obtain the two following minimal sequences solving the CoFaSe problem:

$$000 \xrightarrow{\{d_1^1\}} 101 \xrightarrow{\{d_1^0\}} 011 \xrightarrow{\{d_1^1\}} 110.$$
$$000 \xrightarrow{\{d_1^1\}} 101 \xrightarrow{\{d_1^0\}} 011 \xrightarrow{\{d_0^1\}} 010.$$

Consider now the Boolean network from Figure 3.1 with $S_\alpha = \{010\}$ and $S_\omega = \{100\}$. By following the minimal path between the equivalence classes $[010]$ and $[100]$ in the left graph of Figure 5.2, we obtain the following minimal sequence solving the CoFaSe problem:

$$010 \xrightarrow{\{d_1^2, d_1^3\}} 011 \xrightarrow{\{d_0^2, d_0^3\}} 100.$$

5.3 . Bounds on sequence size for ConEvs dynamics

Determining low upper bounds on the size of control sequences would indicates that algorithms based on a direct exploration of the sequence space can be an efficient solution for sequence inference. Theorem 5.1 establishes such an upper bound, but it is proved for the particular context under which the control can be changed at any state. This bound is not therefore directly applicable to ConEvs, where control changes are only allowed at stable states.

The following theorems establish the upper bounds for this semantics. Theorem 5.2 provides an upper bound of $2^{|\bar{C}_X|}$ under the TCS control application

strategies and Theorem 5.3 provides an upper bound of $2^{|\bar{C}_X|+1} - 1$ under the OCS control application strategies. Thus, the exhaustive exploration of all possible profiles for the C_X-variables should constitute an efficient approach for control sequence computation in comparison with an exhaustive exploration of possible control sequences. For the two theorems, we always consider that every initial state is enduring.

Theorem 5.2. *The size of the minimal contracted total control sequence $\mu_{[k]} \in TCS$ solving the CoFaSe problem (F, S_α, S_ω) for the ConEvs model of dynamics under the synchronous mode is at most $2^{|\bar{C}_X|}$:*

$$\left|\mu_{[k]}\right| \leq 2^{|\bar{C}_X|}.$$

Proof. Consider the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ and assume that $\mu_{[k]} \in$ TCS is a minimal contracted total control sequence solving it for the ConEvs model of dynamics. This control sequence leads to the trajectory $T = s^1 \xrightarrow{\mu_1}{}^* s^2 \ \ldots \ s^k \xrightarrow{\mu_k}{}^* s^{k+1}$, with $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$, and the states s^i, $1 \leq i < k+1$, being the stable states at which the control is changed. We use the symbol τ to refer to the sequence of stable states, plus the initial and the final states: $\tau = (s^i)_{1 \leq i \leq k+1}$.

Now assume that $k > 2^{|\bar{C}_X|}$. Since k is greater than the number of all states over $\bar{C}_X$, it must be that τ contains two states with the same target equivalence class. Assume these two states are at positions $1 \leq i \leq j < k+1$ (*i.e.*, for $[s^i] = [s^j]$).

According to Proposition 5.1, controlling s^i with μ_j reaches the state s^{j+1} in one step. Indeed, s^i and s^j are enduring states; for $s^i \xrightarrow{\mu_j} s'$ and $s^j \xrightarrow{\mu_j} s''$, we have $s'_{\bar{C}_X} = s''_{\bar{C}_X}$. Furthermore, since $[s^i] = [s^j]$, s' is equal to s'', this leads to the trajectory $T' = s^1 \xrightarrow{\mu_1}{}^* s^2 \ \ldots \ s^i \xrightarrow{\mu_j}{}^* s^{j+1} \ \ldots \ s^k \xrightarrow{\mu_k}{}^* s^{k+1}$, contradicting our initial assumption that $\mu_{[k]}$ is minimal.

Therefore, any control sequence solving the CoFaSe problem for the ConEvs model of dynamics with more than $2^{|\bar{C}_X|}$ elements is not minimal, which proves the statement of the theorem. □

Theorem 5.3. *The size of the minimal contracted control sequence $\mu_{[k]} \in OCS$ solving the CoFaSe problem (F, S_α, S_ω) for the ConEvs model of dynamics under the synchronous mode is at most $2^{|\bar{C}_X|+1} - 1$:*

$$\left|\mu_{[k]}\right| \leq 2^{|\bar{C}_X|+1} - 1.$$

Proof. Consider the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ and assume that $\mu_{[k]} \in$ OCS is a minimal contracted control sequence solving it for the ConEvs model of

dynamics. This control sequence gives rise to the trajectory $T = s^1 \xrightarrow{\mu_1}{}^* s^2 \ \ldots \ s^k \xrightarrow{\mu_k}{}^* s^{k+1}$, with $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$ and the states s^i, $1 \leq i < k+1$ being the stable states at which the control is changed. We use the symbol τ to refer to the sequence of stable states, plus the initial and the final states: $\tau = (s^i)_{1 \leq i \leq k+1}$.

Now assume that $k > 2^{|\bar{C}_X|+1} - 1$. Since $k+1$ is greater than double the number of all states over $\bar{C}_X$, it must be that τ contains three states with the same target equivalence class. Assume these three states are at positions $1 \leq h < i < j \leq k+1$ (*i.e.*, for $s^h \xrightarrow{\mu_h} s^{h\prime}$, $s^i \xrightarrow{\mu_i} s^{i\prime}$ and $s^j \xrightarrow{\mu_j} s^{j\prime}$), this means we have $[s^{h\prime}] = [s^{i\prime}] = [s^{j\prime}]$. Note that at least two different controls must appear between the states s^h and s^j because there is at least the stable state s^i in between the two.

By construction the stable states s^i, s^j, and s^h are enduring states, and thus have for target equivalence class their own equivalence class. According to Proposition 5.2, it is possible from s^h to find a control μ'_h reaching s^j in one step. This gives rise to the following trajectory: $T' = s^1 \xrightarrow{\mu_1}{}^* s^2 \ \ldots \ s^h \xrightarrow{\mu'_h} s^j \ \ldots \ s^k \xrightarrow{\mu_k}{}^* s^{k+1}$. This trajectory contradicts our initial assumption that $\mu_{[k]}$ is minimal.

Therefore, any control sequence solving the CoFaSe problem for the ConEvs model of dynamics and with more than $2^{|\bar{C}_X|+1} - 1$ elements is not minimal, which proves the statement of the theorem. □

Theorem 5.2 implies the impossibility of having two states, s^i and s^j, $1 < i < j \leq k+1$, in the contracted trace with identical equivalence class. This point extends to $1 \leq i$ if the initial state is an enduring state. The following corollary formalises this observation:

Corollary 5.1. *Consider a minimal contracted control sequence $\mu^T_{[k]} \in TCS$ solving the CoFaSe problem (F, S_α, S_ω) for the ConEvs model of dynamics under the synchronous mode. Take the sequence $\tau = (s^i)_{1 \leq i \leq k+1}$ of states induced by $\mu^T_{[k]}$, with $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$, and the states s^i, $1 \leq i < k+1$, being the stable states at which the control is changed. In this case, it is impossible to have two indices $1 < i < j \leq k$ such that $[s^i] = [s^j]$.*

Proof. As the two target equivalence classes $[s^i]$ and $[s^j]$ are equal, and as s^j and s^i are enduring states by hypothesis, it follows from Proposition 5.2 and Theorem 5.2 that s^{j+1} can be reached from s^i with the control μ_j. Indeed, s^i and s^j are enduring states for $s^i \xrightarrow{\mu_j} s'$ and $s^j \xrightarrow{\mu_j} s''$, meaning we have $s'_{\bar{C}_X} = s''_{\bar{C}_X}$. Furthermore, since $[s^i] = [s^j]$, s' is equal to s''. Thus, in a minimal

control sequence, it is impossible to have two indices $1 < i < j \leq k$ such that $[s^i] = [s^j]$. □

Duplicates Theorem 5.3 highlights the possibility of the occurrence of states with identical equivalence class in the contracted trace. Since in minimal contracted control sequences all intermediary states are enduring states. The proof of the theorem also entails that *any equivalence class may appear at most twice*. Furthermore, such an appearance necessarily happens in consecutive states of the trace. We refer to these occurrences as *duplicates*. Intuitively, if duplicate equivalence classes appear in non-successive steps i and j, the whole evolution between i and j can be skipped by applying an appropriate control input. These observations exclude the first state as the equivalence class of the first state may appear three times at most. Indeed, if the first state is an impermanent state, then its target equivalence class and its own equivalence class will be distinct. The following corollary formalises this observation:

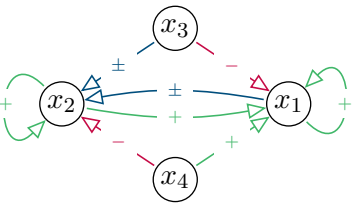
Corollary 5.2. *Consider a minimal contracted control sequence $\mu_{[k]} \in OCS$ solving the CoFaSe problem (F, S_α, S_ω) for the ConEvs model of dynamics under the synchronous mode. Take the sequence $\tau = (s^i)_{1 \leq i \leq k+1}$ of states induced by $\mu_{[k]}$, with $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$, and the states s^i, $1 < i < k+1$, being the stable states at which the control is changed. If there exist two indices $1 < i < j \leq k+1$ such that $s^i \xrightarrow{\mu_i} s^{i\prime}, s^j \xrightarrow{\mu_j} s^{j\prime}$ and $[s^{i\prime}] = [s^{j\prime}]$, then $j = i + 1$.*

Proof. As the two target equivalence classes $[s^{i\prime}]$ and $[s^{j\prime}]$ are equal, and as s^j is an enduring state by hypothesis, it follows from Proposition 5.2 and Theorem 5.3 that it is possible to find a control μ'_i reaching s^j from s^i in one step whenever $j > i + 1$. Thus, in a minimal control sequence, if there exist two indices $1 < i < j \leq k + 1$ such that $s^i \xrightarrow{\mu_i} s^{i\prime}, s^j \xrightarrow{\mu_j} s^{j\prime}$ and $[s^{i\prime}] = [s^{j\prime}]$, then $j = i + 1$. □

For example, the contracted trace of the trajectory 3.7 of the Boolean networks of Figure 3.1 contains the duplicate $x_1 = 0$ in the initial state 010 and the stable state 001.

Examples reaching the bounds. We demonstrate that with the Boolean control networks of Figures 5.3 and Figure 5.4, that the length of the minimal control sequences reaches the bounds outlined by Theorems 5.2 and 5.3.

$$F_{\{u_3^1,u_3^0,u_4^1,u_4^0\}} = \begin{cases} x_1 = x_1 \vee (x_2 \wedge \neg x_3 \wedge x_4) \\ x_2 = (x_1 \wedge x_2) \vee (x_1 \wedge x_3 \wedge \neg x_4) \\ \vee(\neg x_1 \wedge \neg x_3 \wedge \neg x_4) \vee (x_2 \wedge x_3) \\ x_3 = (1 \vee \neg u_3^1) \wedge u_3^0 \\ x_4 = (1 \vee \neg u_4^1) \wedge u_4^0 \end{cases}$$



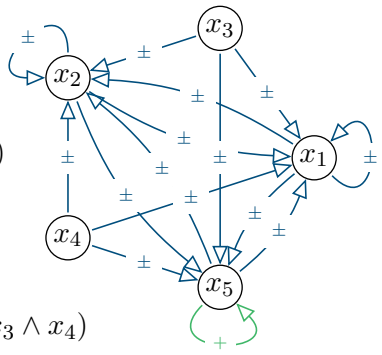
Legend: Boolean control network $F_{\{u_3^1,u_3^0,u_4^1,u_4^0\}}$ with a minimal control sequence reaching the bound defined by Theorem 5.2. The Boolean control network is accompanied by the interaction graph of the corresponding Boolean network without control parameters.

Figure 5.3: Boolean network having a control sequence reaching the bound for TCS ConEvs.

Example 5.1. *In Figure 5.3, in which $\bar{C}_X = \{x_1, x_2\}$, the CoFaSe problem under the ConEvs dynamics in which $S_\alpha = \{0011\}$, $S_\omega = \{1111\}$ has the following TCS sequence of size $2^{|\bar{C}_X|}$ (4) as a solution:*

$$0011 \xrightarrow{\{u_3^0,u_4^0\}}{}^{*} 0100 \xrightarrow{\{u_3^0,u_4^1\}}{}^{*} 1001 \xrightarrow{\{u_3^1,u_4^0\}}{}^{*} 1110 \xrightarrow{\{u_3^1,u_4^1\}}{}^{*} 1111.$$

$$F_{\{u_3^1,u_3^0,u_4^1,u_4^0,u_5^1,u_5^0\}} = \begin{cases} x_1 = (x_1 \wedge \neg x_2) \vee (x_1 \wedge x_3) \vee (x_1 \wedge \neg x_4) \\ \vee(x_1 \wedge x_5) \vee (\neg x_1 \wedge x_2 \wedge \neg x_3 \wedge x_4 \wedge \neg x_5) \\ x_2 = (x_1 \wedge x_2 \wedge \neg x_3) \vee (\neg x_1 \wedge x_2 \wedge x_3) \\ \vee(\neg x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge \neg x_4 \wedge \neg x_5) \vee (x_2 \wedge x_5) \\ \vee(x_1 \wedge \neg x_2 \wedge x_3 \wedge \neg x_4 \wedge \neg x_5) \vee (x_2 \wedge x_4) \\ x_3 = (1 \vee \neg u_3^1) \wedge u_3^0 \\ x_4 = (1 \vee \neg u_4^1) \wedge u_4^0 \\ x_5 = ((x_1 \wedge x_2 \wedge x_3 \wedge \neg x_4) \vee (\neg x_1 \wedge x_2 \wedge \neg x_3 \wedge x_4) \\ \vee(\neg x_1 \wedge \neg x_2 \wedge \neg x_3 \wedge \neg x_4) \vee x_5 \vee \neg u_5^1) \wedge u_5^0 \end{cases}$$



Legend: Boolean control network $F_{\{u_3^1,u_3^0,u_4^1,u_4^0,u_5^1,u_5^0\}}$ with a minimal control sequence reaching the bound defined by Theorem 5.3. The Boolean control network is accompanied by the interaction graph of the corresponding Boolean network without control parameters.

Figure 5.4: Boolean network having a control sequence reaching the bound for OCS ConEvs.

Example 5.2. *In Figure 5.4 in which $\bar{C}_X = \{x_1, x_2\}$ the CoFaSe problem under*

the ConEvs dynamics where $S_\alpha = \{00111\}$, $S_\omega = \{10111\}$has the following OCS sequence of size $2^{|\bar{C}_X|+1} - 1$ (7) as a solution:

$$\begin{aligned} &00111 \xrightarrow{\{u_5^0\}}{}^{*} 00110 \xrightarrow{\{u_3^0,u_4^0\}}{}^{*} 01001 \xrightarrow{\{u_5^0\}}{}^{*} 01110 \xrightarrow{\{u_3^0\}}{}^{*} \\ &11011 \xrightarrow{\{u_5^0\}}{}^{*} 11110 \xrightarrow{\{u_4^0\}}{}^{*} 10101 \xrightarrow{\emptyset}{}^{*} 10111. \end{aligned}$$

6 - Control sequence inference algorithms

Studying the dynamical properties of Total Control Sequence and Open Control Sequence control strategies in Chapter 5 enabled us to define the upper bounds on the length of minimal control sequences. These upper bounds reveal that the uncontrollable variables are central for the inference of a control sequence. As the number of uncontrolled variables is in practice markedly lower than the number of controlled variables (*e.g.*, [7, 11, 13]), the exhaustive exploration of all possible profiles for these variables constitutes an efficient approach for control sequence computation. Thus, we defined algorithms based on partitioning and the exploration of uncontrolled variable profiles. The algorithms infer control minimal sequences that solve the CoFaSe problem for the ConEvs model of dynamics.

In this chapter, we present two algorithms [40, 41] based on $\bar{C}_X$-profiles exploration. In the first section, we present an algorithm for sequence inference. In the second section, we propose the TCS-based inference method, which is less costly but cannot find sequences containing duplicates. In the final section, we present and discuss the benchmarks of the two proposed algorithms.

6.1 . Inference of contracted control sequences

The algorithm inferring a control sequence under the ConEvs model of dynamics builds a tree describing the possible paths reaching a final state of S_ω from the initial set of states S_α. The shortest paths/trajectories are found in this tree, from which the control sequences are then directly derived. By construction, the algorithm avoids redundant operations. Indeed, exploring an already explored $\bar{C}_X$-profile is redundant as all $\bar{C}_X$-profiles reachable from such a profile would have already been reached in the algorithm in one step or with a duplicate.

We impose that our set of initial states S_α will be composed only of stable states of the uncontrolled dynamics of the studied network. This choice allows us to avoid cases in which an initial state belongs to a basin of attraction of a cyclic attractor.[1]

[1]Under the ConEvs model of dynamics, the modification of the control inputs instantiation only occurs at a stable state. If the system is in a cyclic attractor, the control inputs cannot be changed.

Phases of Algorithm 1. Algorithm 1 comprises two major phases.

The first phase (Steps 1 and 2) involves searching for a control allowing to directly attain a state of S_ω.

The second phase (Steps 3 and 4) involves searching for a trajectory visiting an intermediary state within an unexplored equivalence class.

At each step, a parsimonious control input is inferred using the method presented in [7, 8]. The evolution of the main steps is detailed in Figure 6.1.

Data structures. The algorithm relies on the following data structures:

1. *The list* Δ of equivalence classes according to the equivalence relation 5.1.
2. *The exploration tree* G with nodes labelled by sets of stable states and arcs labelled by controls.
3. *The sets* Z_l, Z_{l+1}, *and* Z_{l+2} of unexplored nodes of the tree for the current level of depth l and the next two levels, respectively.

At the beginning, Δ is initialised to $[S_X] \setminus [S_\alpha]$, the depth of $l = 1$, Z_1 contains the root node $\{S_\alpha\}$ of the exploration tree, and all the other data structures are empty. Note that we extend the notation of equivalence class to sets of states: $[A] = \{[a] \mid a \in A\}$.

Duplicate states. A specific treatment is applied to consider the case in which a trajectory passes through duplicates (*i.e.*, the occurrence of states with identical equivalence class). As all states in the explored sequence are stable, and thus enduring states (*i.e.*, their target equivalence class is equal to their equivalence class), a pair of duplicate states can be described as two states belonging to the same equivalence class. By considering that duplicates can only occur in two successive states of any given minimal contracted sequence (Corollary 5.2), we propose two strategies:

1. Let ζ be a set of initial states and ζ' be the set of stable states of F_μ reachable from ζ. then, infer the set of parsimonious control inputs μ validating the following equation:

$$\exists s \in \zeta,\ \exists s' \in S_X : s \xrightarrow{\mu}{}^{*} s' \wedge \mathrm{STBL}_{F_\mu}(s') \wedge [s'] \subseteq [\zeta] \wedge s' \notin \zeta. \qquad (6.1)$$

   Informally, μ is a control that brings the network into a stable state other than the states of ζ, but has the same equivalence class as one of the states

Algorithm 1 Inference of control minimal contracted control sequences

1. *Direct reachability of S_ω:* For all $\zeta \in Z_l$, infer the control μ taking the BCN from ζ to a state in S_ω. If such a μ exists, add the arc labelled by μ to G and go to step 6.

2. *Reachability of S_ω via a duplicate:* For all $\zeta \in Z_l$, infer a pair of controls (μ, μ') such that μ takes the BCN to a state with an equivalence class included in $[\zeta]$, and μ' takes the BCN from there to one of the target state in S_ω. If such a pair of controls exists, add two chained arcs labelled by μ and μ' to G and go to step 6.

3. *Direct reachability of Δ:* For every $\zeta \in Z_l$, infer a set of controls $\mathcal{U}$ taking the BCN from ζ to a state with an equivalence class included in Δ. If $\mathcal{U}$ is non-empty, add the arcs labelled by the controls from $\mathcal{U}$ to G, delete the equivalence class reached in Δ, and store the sets of stable states reached thank to the inferred controls in Z_{l+1}.

4. *Reachability of Δ via a duplicate:* For every $\zeta \in Z_l$, infer a set of pairs of controls $\mathcal{D} = \{(\mu, \mu') \mid \mu, \mu' \in S_U\}$ such that μ takes the BCN to a state with an equivalence class included in $[\zeta]$, and μ' takes the BCN from there to some of the equivalence classes in Δ which could not be directly reached at the previous step. If $\mathcal{D}$ is not empty, add chained arcs labelled by the pairs of controls from $\mathcal{D}$ to G, delete the equivalence class reached in Δ and store the sets of stable states reached thank to the inferred pairs of controls in Z_{l+2}.

5. *Continue if states left:* If one of Z_l, Z_{l+1}, or Z_{l+2} is non-empty, go to step 1 with $l = l + 1$ *i.e.*, discarding Z_l.

6. *Produce the result:* Find the sequence of controls by backtracking G from a leaf found in steps 1 or 2 to the root S_α. If no such leaf was found, return $\emptyset$.

of ζ. For each such control μ, infer a parsimonious control input μ' such that the set of stable states $S_{\mu'}$ of $F_{\mu'}$ reachable from ζ' contains some elements satisfying the desired property (*i.e.*, a state that has the desired equivalence class or a state that belongs to the set of target states S_ω).

2. Let ζ be a set of initial states. then, infer a pair of parsimonious control inputs μ and μ' validating the following equation, with P being the predicate applied to the individual constant x that determines whether x is a state that has the desired equivalence class or a state that belongs to the set of target states S_ω:

$$\begin{aligned}\exists s \in \zeta,\ \exists s', s'' \in S_X : s \xrightarrow{\mu}{}^{*}\ s' \wedge \mathrm{STBL}_{F_\mu}(s') \wedge [s'] \subseteq [\zeta] \\ \wedge\ s' \xrightarrow{\mu'}{}^{*}\ s'' \wedge \mathrm{STBL}_{F_{\mu'}}(s'') \wedge P(s'').\end{aligned} \tag{6.2}$$

We note that in the first strategy, we first infer μ, and then μ', whereas in the second we infer the pair directly. This difference is important as the first strategy may, in some cases, not find the intermediary state needed to reach the desired property. For example, let us consider the following Boolean network:

Example 6.1.

$$F = \begin{cases} x_1 = (x_1 \wedge \neg x_2 \wedge \neg x_3) \vee (\neg x_1 \wedge x_2 \wedge x_3 \wedge x_4) \\ x_2 = \neg x_4 \wedge x_2 \\ x_3 = \neg x_4 \wedge x_3 \\ x_4 = x_4 \end{cases}$$

with $\bar{C}_X = \{x_1\}, C_X = \{x_2, x_3, x_4\}, S_\alpha = \{0000\}$, and $S_\omega = 1\star\star\star$ as CoFaSe parameters. The minimal and control minimal sequence of size 2 reaching a state in S_ω from S_α is $\mu_{[2]} = (\{u_2^1, u_3^1\}, \{u_4^1\})$ inducing the following trajectory:

$$0000 \xrightarrow{\{u_2^1, u_3^1\}} \mathbf{0110} \xrightarrow{\{u_4^1\}} 0111 \xrightarrow{\{u_4^1\}} \mathbf{1001}.$$

Strategy 1 will not find this sequence. In fact, for $s = 0000$ Equation (6.1) only yields the parsimonious controls $(\{u_2^1\}, \{u_3^1\})$, giving the states 0100 and 0010, from which no states reached belongs to the set of target states S_ω.

On the other hand, the second strategy finds all duplicates. In fact, equation (6.2) completely formalises the necessity of having a duplicate to validate the desired property. However, the second strategy is more costly computationally

speaking. In the inference of the pair of controls μ and μ', the dynamics of both F_μ and $F_{\mu'}$ need to be evaluated at the same time, which effectively doubles the number of variables for which an assignment must be found.

Minimality & optimality. The algorithm returns a control minimal sequence (*i.e.*, all control inputs of the contracted sequence are minimal.), but it may be possible to find shorter control sequences.

In the algorithm, the minimality condition is satisfied for the control inputs at each step of the inference. Therefore, for all intermediary states, the returned minimal control is reaching the desired $\bar{C}_X$-profile with a minimal number of perturbations. However, the state reached by applying the minimal control may not be the same as the state that needed to be reached in the $\bar{C}_X$-profile to find the minimal control sequence. In this case, a duplicate is needed to reach such a state. Note that these cases remain rare as they rely on the same complex mechanisms that generate duplicates. Therefore, the algorithm returns a minimal sequence in size when considering the constraint of minimal control to reach the intermediate $\bar{C}_X$-profiles.

Consider the following Boolean network:

Example 6.2.

$$F = \begin{cases} x_1 = x_1 \vee (x_2 \wedge x_3 \wedge \neg x_4) \\ x_2 = (\neg x_1 \wedge x_2) \vee (\neg x_1 \wedge x_4) \vee (x_2 \wedge \neg x_3) \vee (x_2 \wedge x_4) \\ x_3 = (x_1 \wedge x_3) \vee (\neg x_2 \wedge x_3) \vee (x_3 \wedge x_4) \\ x_4 = x_4 \end{cases}$$

with $\bar{C}_X = \{x_1, x_2\}$, $C_X = \{x_3, x_4\}$, $S_\alpha = \{0000\}$, *and* $S_\omega = 11\star\star$ *as Co-FaSe parameters. The control sequence of size* 3 *inferred by Algorithm 1 is* $\mu_{[3]} = (\{u_4^1\}, \{u_3^1\}, \{u_4^0\})$. *However, the following sequence of size* 2*:* $\nu_{[2]} = (\{u_3^1, u_4^1\}, \{u_4^0\})$ *also converges to the same final state* 1100*. Their trajectories are respectively:*

$$\begin{aligned} T_\mu &= 0000 \xrightarrow{\{u_4^1\}} 0001 \xrightarrow{\{u_4^1\}} \mathbf{0101} \xrightarrow{\{u_3^1\}} \mathbf{0111} \xrightarrow{\{u_4^0\}} 0110 \xrightarrow{\{u_4^0\}} \mathbf{1100}, \\ T_\nu &= 0000 \xrightarrow{\{u_3^1, u_4^1\}} 0011 \xrightarrow{\{u_3^1, u_4^1\}} \mathbf{0111} \xrightarrow{\{u_4^0\}} 0110 \xrightarrow{\{u_4^0\}} \mathbf{1100}. \end{aligned}$$

Since the target equivalence class $11\star\star$ *cannot be directly attained from the starting state* 0000*, Algorithm 1 infers a control sequence driving the BCN through some intermediary states. In the example above, the equivalence class* $01\star\star$ *is picked first, and the state* 0101 *is then reached by the single minimal freezing of*

x_4 *to* 1*. The next stable state* 0111*, in the same equivalence class, is reached by minimally freezing* x_3 *to* 1*. The stable state* 1100 *in the target equivalence class is then reached by minimally controlling* x_4 *to* 0*.*

In contrast, applying a non-minimal and non-parsimonious control when considering the constraint of reaching the intermediate $\bar{C}_X$*-profiles results in a smaller sequence. The application of the control* $\{u_3^1, u_4^1\}$ *at the starting state* 0000 *reaches the stable state* 0111 *directly, thereby reducing the length of the control sequence* ν *to* 2*. The first control in* ν *can be considered a union of the first two controls appearing in* μ*, but such a union cannot be generalised.*

Note that the algorithm can be modified in such a way that it returns all the minimal sequences when considering the constraint of minimal control to reach the intermediate $\bar{C}_X$-profiles. In the case where all the returned minimal sequences contain no duplicate, the returned sequences are minimal without considering the constraint on intermediate $\bar{C}_X$-profiles.

Correctness. Algorithm 1 closely follows the proofs of Theorem 5.3 and Corollary 5.2, which guarantees the correctness of the result.

In other words, the sequences found by Algorithm 1 solve a CoFaSe problem for the ConEvs model of dynamics under the synchronous mode, and are control minimal and minimal when considering the constraint of reaching the intermediate $\bar{C}_X$-profiles.

Theorem 6.1. *Algorithm 1 returns all minimal control sequences* $\mu_{[k]}$ *solving the CoFaSe problem for the ConEvs model of dynamics under the synchronous mode when considering the constraint of reaching the intermediate* $\bar{C}_X$*-profiles.*

Proof. The compliance of $\mu_{[k]}$ with the ConEvs model of dynamics and its minimality, when considering the constraint of reaching the intermediate $\bar{C}_X$-profiles, is guaranteed by the use of the minimal one-shot inference algorithm from [7, 8]. This algorithm yields minimal controls, allowing to reach stable states satisfying certain properties. The fact that $\mu_{[k]}$ solves the given CoFaSe problem follows from the end condition in Step 6. □

Example. Figure 6.1 illustrates, on the Boolean control network F_U of Figure 2.7, the computation of a control sequence reaching a state in S_ω from the set of initial states S_α. The data structures are initialised as follows: $\bar{C}_X = \{x_2, x_3\}$, $S_\alpha = \{001\}$, $S_\omega = \{010, 110\}$, $l = 1$, $Z_1 = S_\alpha$, $\Delta = \{11, 10, 00\}$, and all the

Iteration 1:

Direct reachability of S_ω	*Reachability of S_ω via a duplicate*
$\{001\} \longrightarrow s \in S_w$	$\{001\} \longrightarrow [001] \longrightarrow s \in S_w$
Direct reachability of Δ	*Reachability of Δ via a duplicate*
$\{001\} \xrightarrow{(\{u_1^1\})} 11$, 10, 00	$\{001\} \xrightarrow{(\{u_1^1\})} \{111\}$, $[001] \longrightarrow 10$, $[001] \longrightarrow 00$

Iteration 2:

Direct reachability of S_ω
$\{001\} \xrightarrow{(\{u_1^1\})} \{111\} \xrightarrow{(\{u_1^0\})} s \in S_w$

Legend: Steps of the construction of the tree built by Algorithm 1. For the Boolean control network F_U of Figure 2.7 and the set of initial states $S_\alpha = \{001\}$, Algorithm 1 infers the control sequence allowing to reach $S_\omega = \{010, 110\}$. The only controlled variable is $C_X = \{x_1\}$ and the uncontrolled ones are $\bar{C}_X = \{x_2, x_3\}$.
A black arc is a branch of the tree G connecting the previous initial states to the new set of stable states of F_U, under the control given in the annotation. A green arc represents the possibility to reach a state validating the desired property by a set of parsimonious controls. A red arc corresponds to a failure to find a control input leading to the desired property. The returned sequential control is $(\{u_1^1\}, \{u_1^0\})$.

Figure 6.1: Iterations of Algorithm 1 on the Boolean network of Figure 2.7.

other data structures are empty.

1. The first iteration of the algorithm is depicted in four parts that illustrate the construction of the tree during the four phases of the iteration: direct reachability of S_ω, reachability of S_ω via a duplicate, direct reachability of Δ, and reachability of Δ via a duplicate. In this iteration, we first search for a potential trajectory from S_α to the desired property via a direct control or via a duplicate (*i.e.*, a pair of controls validating the Equation 6.1.) In Figure 6.1, we can see that no such trajectories exist.

   As no sequences were found, all equivalence classes in Δ are used as new targets. In Figure 6.1, we see that a state of the equivalence class $[\star 11]$ is reachable from a state of S_α with the control $\{u_1^1\}$, whereas the equiva-

lence classes $[\star 10]$ and $[\star 00]$ are not reachable. Accordingly, we update Δ by deleting 11 and adding to Z_2 the stable states ζ' of $F_{u_1^1}$ belonging to the searched equivalence class $[\star 11]$ and reachable from Z_1. Since in this example, S_α contains only a state and $001 \xrightarrow{F_{u_1^1}}{}^{*} 111$, ζ' will be equal to $\{111\}$.

After creating the new arcs of the tree $s \xrightarrow{\{u_1^1\}} \zeta'$ with $s \in S_\alpha$, we search for new trajectories which would attain an equivalence class in the updated Δ via a duplicate. In Figure 6.1, we see that the equivalence classes $[\star 10]$ and $[\star 00]$ are not reachable via duplicates.

Note that since S_ω contains all the states of the equivalence class $[\star 10]$, trying to reach directly or via a duplicate one of its states is pointless. Indeed, if such action were possible, the algorithm would have stopped at the previous phase. In such a case, deleting the equivalence class from the initial Δ is a simple algorithm optimisation.

2. The second iteration of the algorithm is depicted in Figure 6.1 in one part of which displays only the first phase of the iteration. This iteration of the algorithm is started after updating all the sets of initial variables with new values: $l = 2$, $Z_2 = \{111\}$, and $\Delta = \{00, 10\}$. We can observe, in Figure 6.1, that a state with an equivalence class included in S_ω is reachable from the initial state 111 under the control $\{u_1^0\}$. Since a target state is reached, the main loop of the algorithm stops. After adding the last arc to G, the returned sequential control $(\{u_1^1\}, \{u_1^0\})$ is constructed by backtracking the resulting tree. This sequential control leads to the following trajectory:

$$001 \xrightarrow{\{u_1^1\}}{}^{*} 111 \xrightarrow{\{u_1^0\}}{}^{*} 010.$$

6.2 . TCS-based inference

In this section, we show a two-stage approach for inferring a control sequence: first, infer a total control sequence (TCS) driving the $\bar{C}_X$ subnetwork to satisfy the target property (*i.e.*, reaching a state with the same equivalence class as a state of S_ω); second, derive a control sequence with smaller-size controls (*i.e.*, controls acting on as few control variables as possible). By first inferring a TCS control sequence, we significantly reduce the computation time of the problem for networks with a small number of uncontrolled variables. We will later reveal

that this approach yields solutions of roughly the same quality as those given by Algorithm 1 but is considerably more efficient (Figure 6.3).

Step one: Inference of a TCS sequence of minimal length The first stage of the TCS-based inference is formally described in Algorithm 2, which exhaustively explores all possible $\bar{C}_X$-profiles by building a tree describing the possible paths from the initial states to a target state. As in Algorithm 1, the control sequences are directly derived from this tree by collecting the controls annotating its edges. In contrast to Algorithm 1, TCS inference does not need to explore duplicate states because Theorem 5.2 shows that no duplicate states can occur in minimal TCS sequences. Thus, no solutions are found by Algorithm 2 when the target property can only be reached by control sequences having a duplicate occurrence.

Phases of Algorithm 2. Algorithm 2 comprises two major phases.

The first phase (Step 1) corresponds to the search for a total control allowing to directly attain a state of S_ω.

The second phase (Step 2) corresponds to visiting the intermediary equivalence classes of a given trajectory.

The total controls are inferred with an SAT solver.

Data structures. The algorithm relies on the following data structures:

1. *The list* Δ of equivalence classes according to the equivalence relation 5.1.
2. *The exploration tree* G with nodes labelled by equivalence classes and arcs labelled by total controls.
3. *The sets* Z_l, Z_{l+1} of unexplored nodes of the tree for the current level of depth l, and the next level.

At the beginning, Δ is initialised to $[S_X]/[S_\alpha]$, the depth is $l = 1$, Z_1 contains the root equivalence classes $\{[s] \mid \forall s \in S_\alpha\}$ of the exploration tree, and all the other data structures are empty.

Correctness. Algorithm 2 closely follows the proof of Theorem 5.2, which guarantees the correctness and the minimality of the result. More concretely, the sequences found by Algorithm 2 solve the CoFaSe problem for the totally controlled ConEvs model of dynamics under the synchronous mode. These sequences are minimal.

Algorithm 2 Inference of minimal contracted total control sequences

1. *Direct reachability of* S_ω*:* For all $\zeta \in Z_l$, infer the total control μ^T taking the BCN from the equivalence class ζ to one of the target state in S_ω. If such a μ^T exists, add the arc labelled by μ^T to G and go to step 4.
2. *Direct reachability of* Δ*:* For every $\zeta \in Z_l$, infer a set of total controls $\mathcal{U}^T$ taking the BCN from the equivalence class ζ to some of equivalence classes in Δ. If $\mathcal{U}^T$ is non-empty, add the arcs labelled by the controls from $\mathcal{U}^T$ to G, delete from Δ and store in Z_{l+1} the set of reached equivalence classes.
3. *Continue if states left:* If Z_{l+1} is non-empty, go to step 1 with $l = l+1$ *i.e.*, discarding Z_l.
4. *Produce the result:* Find the trajectory of the sequence of total controls by backtracking G from a leaf found in step 1 to a root equivalence class in Z_0. If no such leaf was found, return $\emptyset$.

Step two: Control minimisation The second stage of the TCS-based inference is formally described in Algorithm 3. The stage reduce a sequence of total controls by testing, for each total control, whether one of its subsets enables reaching the same target equivalence class.

Algorithm 3 comprises only one iterated phase: for every total control μ_i^T appearing in a contracted control trajectory $(s^{i\prime} \xrightarrow{\mu_i^T} s^{i+1\prime})_{1\leq i\leq n}$ induced by a TCS sequence $[\mu_n^T]$, it considers all subsets of μ_i^T and selects the smallest that reaches the same equivalence class as $[s^{i+1\prime}]$. As with all the previous algorithms, Algorithm 3 focuses mainly on the ConEvs model of dynamics.

Data structures. Algorithm 3 relies on the following data structures:

1. *The input contracted total control trajectory* $(s^{i\prime} \xrightarrow{\mu_i^T} s^{i+1\prime})_{1\leq i\leq n}$.
2. *The resulting reduced control sequence* $\mu_{[n]}$.
3. *The sets* S_l, S_{l+1} of unexplored initial states of the reduced sequence for the current level of depth l, and the next level.
4. *The set* S_μ of stable states the BCN can reach under a given control μ.

At the beginning, $(s^{i\prime} \xrightarrow{\mu_i^T} s^{i+1\prime})_{1\leq i\leq n}$ is initialised with a contracted total control trajectory (e.g., one found by Algorithm 2), the depth l is 1, s^1 is equal to S_α, and all the other data structures are empty.

Algorithm 3 Iterative reduction of total controls sequences

1. *Reduce the current total control:* For every control subset μ of the total control μ_l^T, construct the set of stable states S_μ the BCN reaches from S_l under μ. Pick the smallest control μ for which $[s^{l+1\prime}] \in [S_\mu]$. Append μ to $\mu_{[n]}$ and set $S_{l+1} = \{s \mid s \in S_\mu : [s] = [s^{l+1\prime}]\}$.
2. *Continue if the sequence is not finished:* If $l \leq n$, go to step 1 with $l = l + 1$. Otherwise return the reduced sequence $\mu_{[n]}$.

Parsimony & optimality. Algorithm 3 always selects the smallest control subsets. Therefore, if every control in a given TCS sequence is a superset of a corresponding control in a control sequence composed of minimal controls, Algorithm 3 will always find it. However, if the input control sequence is *not* a superset of such a control sequence, the result of Algorithm 3 will be a sequence of parsimonious controls but will not be composed of only minimal controls. In other words, the combination of Algorithm 2 and Algorithm 3 finds parsimonious control sequences.

6.3 . Summary of the algorithms

For reference, we give here a short summary of the approaches to inference and of the algorithms we presented in this section.

The first approach consists of an exhaustive exploration of all possible equivalence classes of $\bar{C}_X$-profiles and controlled trajectories between them. This approach is implemented in Algorithm 1, which organises the explored equivalence classes in a tree, stores the found controls on its arcs, and then derives a control minimal sequence from a path in the tree connecting the root—the starting equivalence classes in $[S_\alpha]$—to the first leaf corresponding to one of the target states in S_ω.

The second approach is also based on exhaustive exploration of equivalence classes and control trajectories, but this time only *total* controls are considered. Algorithm 2 works rather similarly to Algorithm 1: it constructs the tree of explored

equivalence classes, whose arcs are annotated by the found total controls. Since freezing every single C_X-variable is not always necessary, the controls in the output of Algorithm 2 are further individually reduced (Algorithm 3).

6.4 . Benchmarks

To assess the efficiency of the proposed approaches for the inference of control sequences, we benchmarked them on a set of randomly generated Boolean networks. We compared their computational time, memory used and the quality of their results. In the below subsections, we refer to the first approach as *OCS-based inference* and the second approach *TCS-based inference*.

6.4.1 . Experimental protocol

The experiments consist in comparing the effectiveness and the performance of *one-step* control inference by the two proposed approaches on random Boolean networks. Indeed, inferring a single control is the core part of both methods and the principal difference between them. Evaluating the performance of one-step inference is, therefore, a good indicator for comparing their computational costs. Furthermore, considering one-step inference only eliminates the potential bias in the observed resource consumption, which may be introduced by the presence of sequences of different lengths in the sample networks.

As our long-term perspective is to study the controllability of regulatory networks, we generated random networks sharing the same topological propriety. Since it appears that regulatory network topologic is scale-free [1], we generated random Boolean networks from random scale-free interaction graphs obtained using the *Barabási-Albert model* [5]. More precisely, a random Boolean network was randomly generated in two steps from the generated interaction graph:

1. Generation of an undirected graph with respect to a Barabási–Albert graph distribution in which a new node with 3 edges is added at each step with the dedicated function [47] of the Wolfram Mathematica library.

2. Fix the orientation of the generated graph randomly with a probability of self-loop of 0.2 and a probability of generating a positive regulation of 0.6.

We generated networks of 10 to 35 variables with increments of 5. For each sizes, we tested on 100 different random Boolean networks for the inference of a one-step control, reaching a set of target states S_ω from a set of initial states S_α. The initial set of states S_α originated from the set of stable states of the random Boolean network. The initial set of target states S_ω originated from the

enduring states. Around 30% of the network variables were randomly picked to be the uncontrollable variables in $\bar{C}_X$. Table 6.4.1 provides the exact values of different parameters and the technical specifications of the machine used to run the benchmarks.

Characteristics of the computer used to perform the experiments	Model	Macbook Pro				
	CPU	*Intel Core i*7 of 2.8*GHz*				
	RAM	16*Gb* of *DDR*3				
Implementation language	Wolfram Mathematica					
Parameters of the generation of random Boolean networks	*Barabási-Albert model* distribution parameter					2.1
	probability to have a self-loop					0.1
	probability of having positive regulation and not a negative regulation					0.6
Generated Boolean networks size	10	15	20	25	30	35
$\bar{C}_X$-profile size	3	4	6	8	9	10

Table 6.1: Parameters of the benchmarks.

The Boolean formula $f_{x_i}^{n+1}$ is the n-iterated symbolic composition of a variable x_i. It is obtained by replacing all variables in the $f_{x_i}^n$ formula with their respective Boolean formula in the initial controlled Boolean network. Below is an example showing the 3-iterated symbolic composition of the variable x_2 of the controlled Boolean network of Figure 2.7:

$$
\begin{aligned}
f_{x_2}(X,U) &= (x_1 \wedge x_3) \vee (\neg x_1 \wedge x_2) \\
f_{x_2}^2(X,U) &= (f_{x_1}(X,U) \wedge f_{x_3}(X,U)) \vee (\neg f_{x_1}(X,U) \wedge f_{x_2}(X,U)) \\
&= (u_1^0 \wedge \neg u_1^1 \wedge x_1 \wedge x_3) \vee (u_1^0 \wedge \neg x_2 \wedge x_3) \vee (\neg u_1^0 \wedge x_1 \wedge \neg x_3) \\
&\quad \vee (\neg x_1 \wedge \neg x_2) \\
f_{x_2}^3(X,U) &= (u_1^0 \wedge \neg u_1^1 \wedge f_{x_1}(X,U) \wedge f_{x_3}(X,U)) \vee (u_1^0 \wedge \neg f_{x_2}(X,U) \wedge \\
&\quad f_{x_3}(X,U)) \vee (\neg u_1^0 \wedge f_{x_1}(X,U) \wedge \neg f_{x_3}(X,U)) \vee (\neg f_{x_1}(X,U) \wedge \\
&\quad \neg f_{x_2}(X,U)) \\
&= (u_1^0 \wedge \neg u_1^1 \wedge x_1 \wedge x_3) \vee (\neg u_1^0 \wedge x_1 \wedge \neg x_3) \vee (\neg x_1 \wedge \neg x_2) \\
f_{x_2}^4(X,U) &= (u_1^0 \wedge \neg u_1^1 \wedge f_{x_1}(X,U) \wedge f_{x_3}(X,U)) \vee (\neg u_1^0 \wedge f_{x_1}(X,U) \wedge \\
&\quad \neg f_{x_3}(X,U)) \vee (\neg f_{x_1}(X,U) \wedge \neg f_{x_2}(X,U)) \\
&= (u_1^0 \wedge \neg u_1^1 \wedge x_1 \wedge x_3) \vee (\neg u_1^0 \wedge x_1 \wedge \neg x_3) \vee (\neg x_1 \wedge \neg x_2)
\end{aligned}
$$

Note that in the example $f^3_{x_2}(X,U) = f^4_{x_2}(X,U)$. This is a fixed point in the composition. Since the composition of x_2 reached a fixed point, we know that a state s will reach a state s' of an attractor of the dynamics of F_μ in which $s'_{x_2} = f^3_{x_2}(s,\mu)$.

The Boolean formula encapsulating the synchronous reachability is obtained by a logical conjunction of the 20-iterated symbolic composition of all variables $x_i \in X$. Preliminary experiments revealed that 20 compositions were enough to reach stable states in the sequence inferring algorithms.

The inference of one-step controls is time-constrained to 1 *hour* of computation time. Due to excessive computational time of OCS-based inference, the method is only applied to networks of size 10 and 15 (see Figure 6.2).

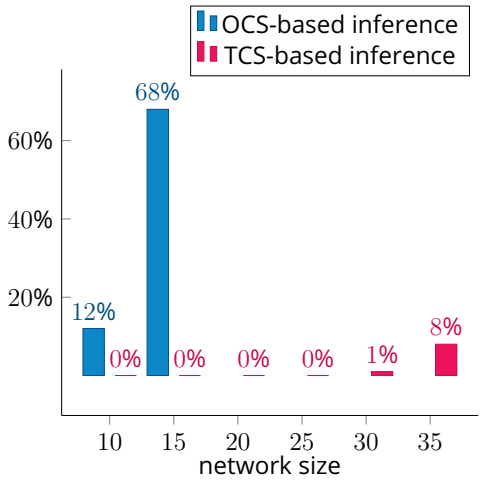


Figure 6.2: Percentage of the OCS-based and TCS-based one-step inference runs, aborted after an hour of computation.

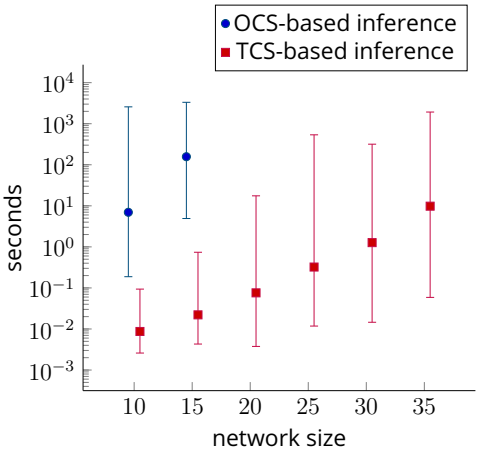


Figure 6.3: Median of the computation time of the OCS-based and TCS-based one-step inference, excluding aborted runs.

6.4.2 . Description of the resulting data

The figures, contain the median values for multiple computations since the outliers present in the dataset risk biasing the average.

Figure 6.2 describes the percentage of timed-out computations. Figures 6.3, 6.4, and 6.6 are realised without considering the aborted inferences. Figure 6.3 illustrates the median of the total computational time of the two approaches, including the minimum and the maximum. Figure 6.4 displays the median, with the minimum and the maximum, of the total memory used in bytes of the two

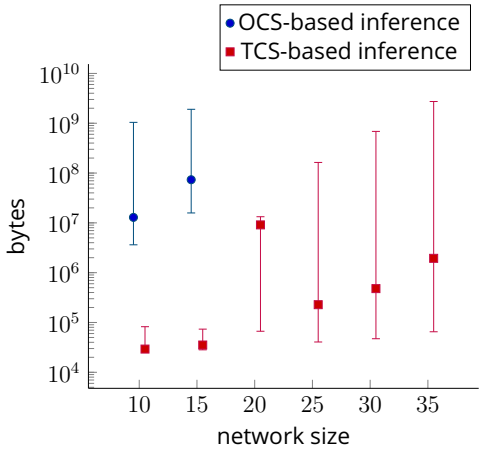


Figure 6.4: Median of memory usage by OCS-based and TCS-based one-step inference, excluding aborted computations.

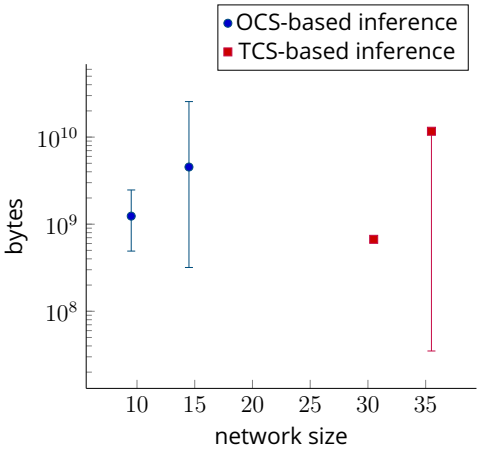


Figure 6.5: Median of memory usage by OCS-based and TCS-based inferences in aborted computations.

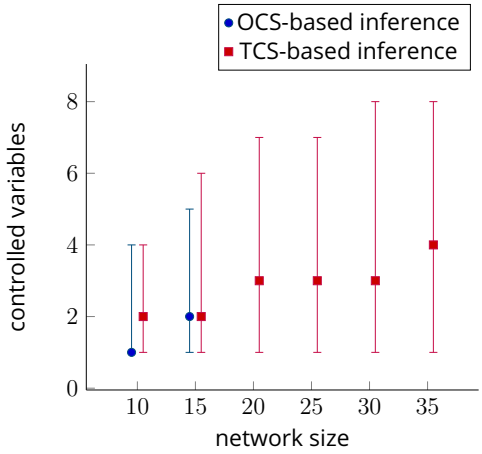


Figure 6.6: Median of the sizes of controls found by the OCS-based and TCS-based one-step inference, excluding aborted computations.

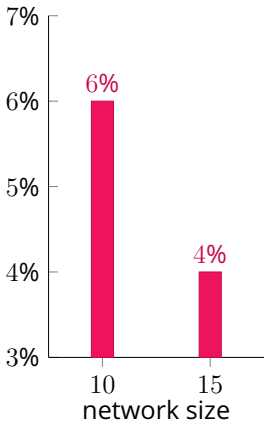


Figure 6.7: Percentage of times OCS-based inference yields a smaller control than TCS-based inference.

approaches. Figure 6.5 describes the median, with the minimum and the maximum, of the total memory used in bytes by the timed-out inferences. Figure 6.6 illustrates the median, with the minimum and maximum, of the number of variables controlled by the inferred one-step controls. Figure 6.7 displays the percentage of times the TCS-based inference finds a parsimonious but not optimal control (*i.e.*, the inferred control is larger than the OCS-based inferred control.) We recall that as OCS-based inference is optimal by construction, a control inferred by the TCS-based inference cannot be smaller than the corresponding OCS-based inferred control.

6.4.3 . Discussion of the benchmarks

The benchmark clearly shows a performance gap between the computation time of the two approaches (Figure 6.3). This disparity becomes more obvious when taking into account that for networks of sizes 10 and 15, respectively 10% and 70% (Figure 6.2) of computations are aborted after 1 hour, and thus are omitted in Figure 6.3.

Figure 6.2 reveals that, on networks of size 35, the TCS-based inference resulted in a similar number of aborted computations as the OCS-based inference for networks of size 10. Furthermore, Figure 6.3 shows that the times of the TCS-based inference on networks of size 35 are similar to the times of the OCS-based inference on networks of size 10. This correspondence is due to the fact that the TCS-based inference discards the dynamics of the controlled variables by freezing all of them. Since we selected around one-third of variables to be uncontrollable, the TCS-based inference considered the dynamics of only about 10 variables for networks of size 35.

In Figure 6.4, we observe a jump in median memory usage for networks of size 20. As minimum and maximum values seem within ranges that agree with the other bounds visible on the graph, this variation is probably random and unrelated to the intrinsic properties of the algorithm.

In Figures 6.4 and 6.5, for networks of size 10 and 35, we observe the same pattern as in Figures 6.2 and 6.3 while noting that the median memory used by the OCS-based inference is larger than its counterpart. This outcome can be attributed to the fact that Boolean formulas, for the reachability of the TCS-based inference, may generally be less complex for similar numbers of variables because the sets of $\bar{C}_X$-variables are randomly selected. Therefore, the $\bar{C}_X$-variables subnetwork does not follow the power-law distribution of scale-free networks. This difference should result in a network with fewer arcs than a scale-free network with a similar number of variables. In Figure 6.6, the size of inferred controls remains small, even when the size of the network increases. Therefore, we can conclude that the reduction

carried out by Algorithm 3 generally results in rather small controls.

Since the size of the Boolean formula for reachability increases as the network size increases, the time needed for the inference of a one-step control by the solver also increases. By only considering the dynamics of $\bar{C}_X$-profiles, the TCS-based inference postpones the computational explosion, in contrast to the OCS-based inference. As we can see in all the figures, this entails a better performance in terms of time and memory used by the second approach compared with the first one. In Figure 6.7, the number of times where the TCS-based algorithm provides a parsimonious but not optimal solution is low. However, given the low number of instances where both algorithms returned a result (120 instances in total), this observation should be taken with caution.

In these benchmarks, we compared the effectiveness and the performance of one-step control inference of the OCS-based approach and the TCS-based approach on random Boolean networks. From these experiments, we observe that TCS-based inference, in comparison with the OCS-based inference, is less costly in computation time and memory usage. By definition, the OCS-based inference always returns the minimal control. By comparing the instance where the two algorithms return a result, we note that TCS-based inference seems to return minimal controls most of the time. From these observations, we conclude that the TCS-based approach infers solutions of good quality and is a rather viable alternative to the more computationally costly OCS-based approach.

7 - Structural sequence analysis

By relying on knowledge about the topology, it would be possible to use the sequence bound parameter as well as other topological properties to select fewer $\bar{C}_X$-profiles to explore in order to reduce the computational cost of the inference algorithm. The previous chapters proved an upper bound of the sequence with regard to the number of uncontrolled variables. In this chapter, we examine the determination of this bound according to the topology of the network. Indeed, we can establish a tighter upper bound for some topologies. The goal is to establish some causal relationship between the topology of the network and the control.

The approach presented in Section 5.3 offers a solution based on the dynamical properties of TCS and OCS control strategies. The proposed methodologies explore the set of $\bar{C}_X$-profiles of the controlled Boolean network to infer a sequence of controls. To limit the number of $\bar{C}_X$-profiles to explore, we are particularly interested in eliminating profiles that cannot have an impact the course of a sequence of controls. For example, $\bar{C}_X$-profiles containing no stable states do not need to be visited under the ConEvs model of controlled dynamics because a control change can only occur in a stable state.

We seek to answer the following question: '*Given a network topology, what is the upper bound of the control sequence minimal size?*'

A controlled Boolean network is a function generating a set of Boolean networks with different dynamics. Thus, the dynamical behaviour of the Boolean network greatly depends on its control inputs. In controlled Boolean networks, $\bar{C}_X$-variables cannot be controlled. Their update functions are, therefore, the same regardless of the control. Since possible controls include total controls in which all C_X-variables are controlled to '0' or '1', the properties that imply the necessity of a sequence of control should emerge from the update functions of $\bar{C}_X$-variables. As the interaction graph is a compact and static abstraction of Boolean network dynamics [45], studying the topology of the signed interaction graph of the $\bar{C}_X$-variables should offer insights into which dynamical interactions between $\bar{C}_X$-variables play a central role in the CoFaSe problem.

In this chapter, we formally describe the relationship between structure and the

length of control sequences, and we provide the corresponding theorems. In the first section, we study the evolution of the interaction graph as a function of the control. We reveal that the only important part corresponds to the $\bar{C}_X$-subgraph. Based on this subgraph, we study the bounds on sequence length. Section 7.2, we study the structural properties of non-negative cycles. Finally, in Section 7.3, we present the new set of bounds.

7.1 . Impact of the interaction graph on the control

The goal of this section is to study the link between the sequential control and the interaction graph. We begin in the first subsection by presenting the $\bar{C}_X$-interaction graph which is a subgraph that remains invariant for all Boolean networks generated by the BCN. We conclude in the second subsection by describing the structural condition for the emergence of sequential control. This structural condition is the necessity of the presence of at least a positive loop in the $\bar{C}_X$-interaction graph for a minimal sequence of control to be of a size greater or equal to two.

7.1.1 . $\bar{C}_X$-interaction graph

A Boolean control network is a family of Boolean networks parameterised by the control input. Each valuation of the control, therefore, leads to a specific Boolean network. As an interaction graph is the structural representation of a Boolean network, the interaction graph is thus also modified by the control. Although the interaction graphs may differ for the generated Boolean networks, only the update functions of the C_X-variables may be altered. Indeed, by construction, the update functions of $\bar{C}_X$-variables are not directly affected by a modification in control since they cannot be controlled. *Thereby, whatever the control applied to a Boolean control network, the interactions between $\bar{C}_X$-variables do not vary*. The $\bar{C}_X$-subgraph will be the same for any Boolean network generated by a Boolean control network with a given set of $\bar{C}_X$-variables. We denote this subgraph the $\bar{C}_X$-interaction graph. This representation is central for the structural analysis of the sequence bound.

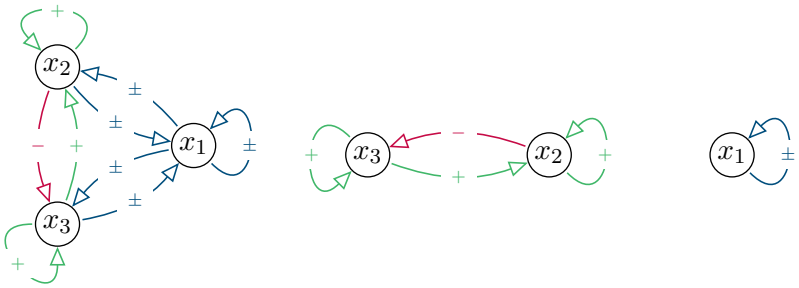
$\bar{C}_X$-interaction graph. The $\bar{C}_X$-interaction graph of a Boolean control network captures the interdependence of the $\bar{C}_X$-variables of the network occurring in the dynamics. This subgraph is a directed graph $\langle \bar{C}_X, \longrightarrow\!\!\triangleright \rangle$.Strictly speaking, a BCN does not have an interaction graph, but since $\bar{C}_X$-variables are not controlled,

the interactions between these variables remain the same regardless of the control input.

The $\bar{C}_X$-interaction graph can be constructed from the signed interaction graph of the Boolean network without the control parameter since its interaction graph corresponds to the interaction graph of the Boolean network yielded by the empty control.

Figure 7.1 illustrates the signed interaction graph of a Boolean network f of Figure 2.1 with the signed $\bar{C}_X$-interaction graph of two controlled Boolean networks with a different set of controllable variables on F:

- The BCN of Figure 2.7, $F_{\{u_1^0,u_1^1\}}$ in which the variable x_1 is controlled
- The BCN of Figure 3.1, $F_{\{u_2^0,u_2^1,u_3^0,u_3^1\}}$ in which the variables x_2 and x_3 are controlled.



Signed interaction graph of F | Signed $\bar{C}_X$-interaction graph of $F_{\{u_1^0,u_1^1\}}$ | Signed $\bar{C}_X$-interaction graph of $F_{\{u_2^0,u_2^1,u_3^0,u_3^1\}}$

Legend: F is the Boolean network of Figure 2.1. On the left is the signed interaction graph of F. In the middle is the signed $\bar{C}_X$-interaction graph of $F_{\{u_1^0,u_1^1\}}$ where the unique controllable variable is x_1 and the uncontrollable variables are x_2 and x_3. On the right is the signed $\bar{C}_X$-interaction graph of $F_{\{u_2^0,u_2^1,u_3^0,u_3^1\}}$ where the controllable variables are x_2 and x_3 and the unique uncontrollable variable is x_1.

Figure 7.1: The signed $\bar{C}_X$-interaction graph of Boolean control networks.

7.1.2 . Necessary conditions for sequences

We want to find the structural condition for the emergence of sequential control. In this section, we show that the size of the sequence depends on the structure of the $\bar{C}_X$-interaction graph. More precisely, we explain the necessity of a non-negative cycle in the $\bar{C}_X$-interaction graph to have non-trivial control sequences (*i.e.*, of size greater or equal to two).

One of the central conditions for the existence of non-trivial control sequences is that all controlled Boolean networks validating the target property[1] on an attractor must have at least two basins of attraction. Indeed, no control allows to reach the property from the initial states. If this were the case, the sequence would be of size one. Moreover, under ConEvs dynamics, the property must be reached by a stable state.

Thus, since all Boolean networks necessarily fall into an attractor, the target property cannot be satisfied in all the reachable attractors from the set of initial states. Therefore, In the case of a minimal sequence of control of size greater than or equal to two, in all the Boolean networks produced by the controlled Boolean network and in which the property is validated by one of its stable states, there are at least two attractors: the one that satisfies the property an cannot be attainable from all states of S_α, and the others.

Since the above sentence is true for all control inputs, it is also true for total control (*i.e.*, control in which all C_X-variables are controlled to 1 or 0). In this case, the fact that the control is total means that the attractors of the generated Boolean network must be generated by its set of $\bar{C}_X$-variables.

Note that for all controlled BCNs validating the target property on one of their stable states denoted s, there exists a total control in which all C_X-variables are controlled to have the same instantiation as in s, which results in a Boolean network having s as a stable state.

Thus, in the case of a minimal sequence of control of size greater than or equal to two, in all the Boolean networks produced by the controlled Boolean network, and in which the property is validated by one of its stable states, there must exist a total control also validating the property on one of its stable states and in which there are at least two attractors: the one attainable from all states of S_α and the other that satisfies the property. If this were not true, all control sequences would

[1] A state that has a desired equivalence class or a state that belongs to the set of target states S_ω.

be trivial and be reached in one step with total control. Therefore, the situations in which one-step controls are insufficient could be explained by the interactions between $\bar{C}_X$-variables.

We now seek to identify which structures generate multiple basins of attraction in the dynamics of a Boolean network. We then transpose these structures into a $\bar{C}_X$-interaction graph and observe what is their influence on sequences of control.

In Boolean networks, complex behaviours in dynamics require the presence of cycles [48]. To understand the influence of such cycles, we rely on the following results:

Theorem 7.1 (Robert [52])**.** *Let F be a Boolean network and assume that the interaction graph of F is acyclic. Then, F in synchronous and asynchronous modes has a unique attractor which is a fixed point.*

Theorem 7.2 (Aracena [3])**.** *Let F be a Boolean network and assume that the interaction graph of F is strongly connected, has at least one arc, and has no positive cycle. Then, F in synchronous modes has no fixed points.*

According to Theorems 7.1 and 7.2, for a Boolean network to have a fixed point and more than one attractor in its dynamics, its interaction graph must not be acyclic or a strongly connected component containing no positive cycles.

Relaying on Theorems 7.1 and 7.2, the following two theorems emphasise the importance of non-negative cycles in sequences of control. The following folklore theorem states that the interaction graph of a Boolean network must contain a non-negative cycle in order to have in the dynamics at least two attractors, among which is at least one fixed point.

Theorem 7.3 (Folklore)**.** *Let F be a Boolean network and assume that the interaction graph of F has no non-negative cycle. Then, in synchronous modes, F cannot have two attractors among which at least one is a fixed point.*

Proof. Consider the Boolean network F in which the interaction graph of F has no non-negative cycle. We assume, for the sake of contradiction, that F has at least a fixed point attractor and another attractor. Thus, the interaction graph of F cannot be acyclic and must contain a negative cycle (Theorem 7.1). According to the definition of a negative cycle, the interaction graph of F must contain at least a strongly connected component. Thus, we can define:

- B as the set of variables belonging to complex strongly connected components in F.

- $B^{\swarrow}$ as the set of upstream variables of B.

For F to have a fixed point, $B^{\swarrow}$ must not be empty. Indeed, since F has a fixed point, all complex SCC composed of negative cycles must be stabilised to a fixed point by a set of upstream variables (Theorem 7.2). We can then define $F_{B^{\swarrow}}$ as the subnetwork composed of the variables of $B^{\swarrow}$. By definition, there cannot be any arcs going from B to the variables of $B^{\swarrow}$, which means that the update functions for the variables of $B^{\swarrow}$ do not depend on B, and therefore taking them separately provide a perfectly valid Boolean subnetwork. $F_{B^{\swarrow}}$ must have at least two fixed points such that one instantiation of the variables of $B^{\swarrow}$ stabilises all complex strongly connected components to a fixed point, and another to destabilise at least one complex strongly connected component to a cyclic attractor or stabilises all complex strongly connected components to another fixed point. This is impossible because the interaction graph of $F_{B^{\swarrow}}$ is by definition acyclic (Theorem 7.1), thus validating the above theorem. □

The presence of non-negative cycles in the $\bar{C}_X$-interaction graph is crucial for the emergence of sequential control. We observe that the presence in the $\bar{C}_X$-interaction graph of positive cycles and positive/negative cycles must play a preponderant role in the emergence of sequences of control. For example, all the previously presented Boolean networks with CoFaSe problems solved by minimal sequences of control of size greater than or equal to two, $F_{\{u_1^0,u_1^1\}}$ with the sequence 3.6, and $F_{\{u_2^0,u_2^1,u_3^0,u_3^1\}}$ with the sequence 3.7, have at least a non-negative cycle in their respective signed $\bar{C}_X$-interaction graphs (Figure 7.1). The following theorem confirms this central role. *It states that without a non-negative cycle in the $\bar{C}_X$-interaction graph, all properties reachable with a control or sequence of controls will be reachable with a one-step control strategy.*

Theorem 7.4. *Let F_U be a Boolean control network. The existence of a sequence of minimal size $\mu[k]$ with $k \geq 2$, solving the CoFaSe problem under the ConEvs model of dynamics and synchronous modes implies the presence of at least a non-negative cycle in the $\bar{C}_X$-interaction graph of F_U.*

Proof. Consider the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ and assume that $\mu_{[k]}$ with $k \geq 2$ is a minimal contracted total control sequence solving the problem for the ConEvs model of dynamics. This control sequence gives rise to the trajectory $T = s^1 \xrightarrow{\mu_1}{}^* \ldots \xrightarrow{\mu_k}{}^* s^{k+1}$, with $s^1 \in S_\alpha$ and $s^{k+1} \in S_\omega$.

As s^{k+1} is a stable state, it is possible to find a total control μ^T such that $\text{stbl}_{F_{\mu^T}}(s^{k+1})$. Since $\mu_{[k]}$ is minimal, $s^1 \xrightarrow{\mu^T}{}^*$ does not reach the fixed point

s^{k+1}. Thus, the dynamics graph of F_{μ^T} must have at least a fixed point and another attractor. According to Theorems 7.1 and 7.3, for F_{μ^T} to have at least a fixed point and another attractor, its interaction graph must contain at least one cycle that is not negative. Moreover, since μ^T is a total control, the cycle must be enclosed in the $\bar{C}_X$-interaction graph of F_U as all C_X-variables are constant. □

7.2 . Cycle structural properties

Finding what attractors can be generated by the application of the different control inputs of a Boolean control network is crucial for understanding the causality of the CoFaSe problem. Theorem 7.4 explains that such a basin of attractions must arise from non-negative cycles present in the $\bar{C}_X$-interaction graph.

In this section, we focus on studying the structural properties of single non-negative cycles. This section is a technical section used to define new notions that solve the proofs of the new set of tight upper bounds (Section 7.3). In Subsection 7.2.1, we discuss what stable states an isolated positive cycle can have. In Subsection 7.2.2, we explore the effect of the upstream variables of a cycle on its dynamics. Finally, in Subsection 7.2.3, we formalise some of the structural and dynamical properties of non-negative cycles.

7.2.1 . The stable states of a positive cycle

We condiser the stable states that an isolated positive cycle can have and whether they have a particular property. Several structural properties have already been obtained by Robert [49, 51, 52], Shih and Dong [54], Aracena [3, 4], Goles [26], Demongeot [18], and Richard [48]. For the particular case of positive cycles, Aracena [3] proved the following theorem.

Theorem 7.5 (Aracena [3])**.** *Let F be a Boolean network. Assume that its interaction graph is strongly connected, has at least one arc, and has no negative cycles. Then, F has at least two attractors and two fixed points of opposite instantiations $s = \neg s'$.*

This theorem highlights the presence of two stable states of opposite instantiations in the dynamics of Boolean networks for interaction graphs consisting of a strongly connected component without negative cycles. However, this theorem is too general and encompasses more than a single positive cycle. We need to know exactly the nature and the number of fixed point attractors a positive cycle can generate.

From the work of Goles in [26], it can be found that an isolated positive cycles trivialy have two fixed points. This observation was made by Demongeot in [18] and provides the following lemma:

Lemma 7.1. *Let F be a Boolean network. Assume that its interaction graph consists of a unique positive cycle. Then, F has at least two attractors and exactly two fixed points of opposite instantiations $s = \neg s'$.*

Proof. This lemma is a weakening of the number of p-attractors of a positive Boolean cycle given in Theorem 1 of [18]. □

Lemma 7.1 can be applied to a cycle without upstream in a larger interaction graph. Since, in such a network, $\mathbb{c}_i$ has no upstream variables, we can study the evolution of the values of the variables of $\mathbb{c}_i$ independently of the dynamics of the other variables. Concretely, we examine the projections of the trajectories of the network on the variables of $\mathbb{c}_i$. and notice that $\mathbb{c}_i$ can be in exactly two stable states with opposite instantiations.

Opposite instantiations of a cycle. We denote the two opposite instantiations of a cycle $\mathbb{c}_i$ as $\gamma^{\top}_{\mathbb{c}_i}$ and $\gamma^{\bot}_{\mathbb{c}_i}$. We also denote the set of states in which $s''_{\mathbb{c}_i} \neq \gamma^{\top}_{\mathbb{c}_i} \wedge s''_{\mathbb{c}_i} \neq \gamma^{\bot}_{\mathbb{c}_i}$ as $\Gamma^{\odot}_{\mathbb{c}_i}$ *i.e.*, $\Gamma^{\odot}_{\mathbb{c}_i}$ is the set of instantiation of $\mathbb{c}_i$ variables that are neither $\gamma^{\top}_{\mathbb{c}_i}$ nor $\gamma^{\bot}_{\mathbb{c}_i}$. We also denote as $\gamma^{\odot}_{\mathbb{c}_i}$ any element of $\Gamma^{\odot}_{\mathbb{c}_i}$.

7.2.2 . Effects of the upstream variables of a cycle

In this subsection, we study the effect on a cycle of the instantiation of its upstream variables. For some upstream variables instantiation, a cycle can be considered to behave like a *virtual cycle* whit no upstream variables. If a virtual cycle is a positive cycle, Lemma 7.1 applies.

The instantiation of the upstream variables of a cycle can have two effects on it. To explain these effects, we define for a given cycle $\mathbb{c}$ the following sets of variables:

- $\{y_1, \ldots, y_k\} = \mathbb{c}$: the set of variables of the cycle,
- $\{z_1, \ldots, z_l\} = X \setminus \mathbb{c}$: the set of variables not belonging to the cycle.

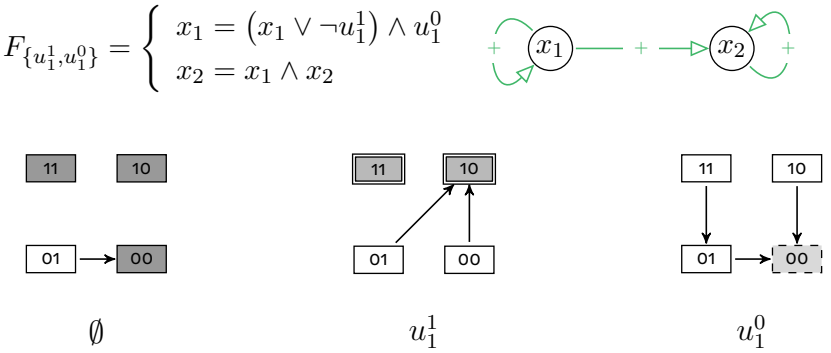
We denote $\Phi(f)$ as the minimal *DNF* of a Boolean formula f. A cycle $\mathbb{c}$ may have the following:

- A set of *non-absorbing instantiations* $N_{\mathbb{c}}$ of its upstream variables in which the instantiation of the cycle can be considered to behave like a *virtual cycle* with no upstream variables.

$$\begin{aligned} &\forall s \in S_X : s_{\mathbb{c}^{\swarrow}} \in N_{\mathbb{c}} \implies \\ &\forall y_i \in \mathbb{c} : \Phi(f_{y_i}(y_1, \ldots, y_k, s_{z_1}, \ldots, s_{z_l})) \notin \{1, 0\}. \end{aligned} \tag{7.1}$$

- A set of *absorbing instantiations* $A_{\mathbb{c}}$ of its upstream variables in which the instantiation of the cycle directly depends on its upstream variables.

$$\begin{aligned} &\forall s \in S_X : s_{\mathbb{c}^{\swarrow}} \in A_{\mathbb{c}} \implies \\ &\exists y_i \in \mathbb{c} : \Phi(f_{y_i}(y_1, \ldots, y_k, s_{z_1}, \ldots, s_{z_l})) \in \{1, 0\}. \end{aligned} \tag{7.2}$$



Legend: The synchronous dynamics of a Boolean control network $F_{\{u_1^1, u_1^0\}}$ with its interaction graph. From left to right, the respective controls are: no freeze, x_1 is frozen to 1, x_1 is frozen to 0. The active control parameters are mentioned below each dynamics. The stable states of each dynamics are coloured in three shades of grey, and their contours are drawn in different styles. Each contour style is associated with a different control input.

Figure 7.2: The synchronous dynamics of a BCN used to illustrate non-absorbing and absorbing instantiations.

For example, take the Boolean controlled network of Figure 7.2 in which $C_X = \{x_1\}$ and $\bar{C}_X = \{x_2\}$. In this case, the $\bar{C}_X$-interaction graph consists of a unique positive cycle $\mathbb{c} = \{x_2\}$, which has for upstream variables the set $\mathbb{c}^{\swarrow} = \{x_1\}$. $\{x_1 = 0\}$ is an absorbing instantiation of the upstream variable of $\mathbb{c}$ as $f_{x_2} = 0 \wedge x_2 = 0$. For any control μ in which x_1 is equal to 0, x_2 is equal to 0 at

the next dynamics step (*i.e.*, $\forall \mu \in S_U : 0\star \xrightarrow{\mu} 0\star$).[2] On the other hand, $\{x_1 = 1\}$ is a non-absorbing instantiation of the upstream variable of $\mathbb{c}$ as $f_{x_2} = 1 \wedge x_2 = x_2$. For any control, when x_1 is equal to 1, x_2 is equal to itself at the next dynamics step (*i.e.*, $\forall \mu \in S_U, \alpha \in \mathbb{B} : 1\alpha \xrightarrow{\mu} \star\alpha$).

7.2.3 . Cycle dynamics

In this subsection, we formalise the structural properties of the non-negative cycles when their upstream variables are set to non-absorbing instantiations. If a stable state has the upstream variables of the cycle $\mathbb{c}$ instantiated to one of its non-absorbing instantiations, then this state only has a limited set of possible instantiations for the variables of the cycle $\mathbb{c}$. Indeed, $\mathbb{c}$ can be considered for each non-absorbing instantiation to have the dynamics of a *virtual cycle* denoted $\mathfrak{c}$ without any upstream variables and with the same set of variables as the cycle $\mathbb{c}$. Lemma 7.1 can be applied to the subgraph composed of cycle $\mathfrak{c}$. We define for a given cycle $\mathbb{c}$ the following sets of variables:

- $\{y_1, \ldots, y_k\} = \mathbb{c}$: the set of variables of the cycle.
- $\{z_1, \ldots, z_l\} = X \setminus \mathbb{c}$: the set of variables not belonging to the cycle.

The functions of the Boolean network f', which has for interaction graph the subgraph composed of the cycle $\mathfrak{c}$, are defined as follows:

$$\begin{aligned} &\forall s \in S_X : s_{\mathbb{c}^{\swarrow}} \in N_{\mathbb{c}} \implies \\ &\forall y_i \in \mathfrak{c} : f'_{y_i} = \Phi(f_{y_i}(y_1, \ldots, y_k, s_{z_1}, \ldots, s_{z_l})). \end{aligned} \tag{7.3}$$

Since the Boolean network f' has for interaction graph a unique cycle $\mathfrak{c}$, each function of its variables has exactly one parameter. Thus, $\mathfrak{c}$ has '$\pm$' arcs as such arcs require a function with at least two input parameters.

Therefore, for a state s in which $s_{\mathbb{c}^{\swarrow}} \in N_{\mathbb{c}}$, the non-negative cycle $\mathbb{c}$ can be considered to have the dynamics of a positive or a negative cycle $\mathfrak{c}$. We now introduce the following notations:

- For each pair $\{y_j, y_i\}$ of variables of $\mathbb{c}$ in which $y_j \xrightarrow{+} y_i$, the relation from y_j to y_i in $\mathfrak{c}$ is labelled by 1 ('$+$').
- For each pair $\{y_j, y_i\}$ of variables of $\mathbb{c}$ in which $y_j \xrightarrow{-} y_i$, the relation from y_j to y_i in $\mathfrak{c}$ is labelled by -1 ('$-$').

[2] $0\star$ and $\star 0$ respectively correspond to the sets of states $\{00,\ 01\}$ and $\{00,\ 10\}$.

- For each pair $\{y_j, y_i\}$ of variables of $\mathbb{c}$ in which $y_j \xrightarrow{\pm} y_i$, the relation from y_j to y_i in $\mathfrak{c}$ is labelled by 1 ('$+$') or -1 ('$-$')[3].

From these observations, we can refine the Equation 7.1 as follows:

$$\begin{aligned}
&\forall s \in S_X : s_{\mathbb{c}\swarrow} \in N_{\mathbb{c}} \implies \forall y_i, y_j \in \mathbb{c} : \\
&(y_j \xrightarrow{+} y_i \implies \Phi(f_{y_i}(y_1, \ldots, y_k, s_{z_1}, \ldots, s_{z_l})) = y_j) \wedge \\
&(y_j \xrightarrow{-} y_i \implies \Phi(f_{y_i}(y_1, \ldots, y_k, s_{z_1}, \ldots, s_{z_l})) = \neg y_j) \wedge \\
&(y_j \xrightarrow{\pm} y_i \implies \Phi(f_{y_i}(y_1, \ldots, y_k, s_{z_1}, \ldots, s_{z_l})) \in \{y_j, \neg y_j\}).
\end{aligned} \tag{7.4}$$

The following proposition helps reveal on the nature of the set of possible instantiations of variables of a cycle $\mathbb{c}$ in which a stable state has the upstream variables of $\mathbb{c}$ instantiated to one of its non-absorbing instantiations. The proof is based on the observation that in this situation the virtual cycle of $\mathbb{c}$ must be positive.

Proposition 7.1. *Let $\mathbb{c}$ be a cycle in the interaction graph of a Boolean network F. For all stable states s of F where $s_{\mathbb{c}\swarrow} \in N_{\mathbb{c}}$, $s_{\mathbb{c}}$ is only equal to one of the opposite instantiations of one of its positive virtual cycles.*

Proof. Let $\mathbb{c}$ be a cycle in the interaction graph of a Boolean network. For all states s where $s_{\mathbb{c}\swarrow} \in N_{\mathbb{c}}$, $\mathbb{c}$ will behaves like a virtual cycle $\mathfrak{c}$. When s is a stable state, $\mathfrak{c}$ must be a positive cycle. Indeed, since each function of the variables of the resulting virtual cycle $\mathfrak{c}$ has exactly one parameter, this cycle cannot have '$\pm$' arcs. Thus, $\mathfrak{c}$ can only be a positive or negative cycle. If $\mathfrak{c}$ was negative, s would not be a stable state as the variable of $\mathbb{c}$ would have the dynamics of a negative cycle, which that cannot have a stable state (Theorem 7.2). Moreover, if s is a stable state, $s_{\mathbb{c}}$ must have the same instantiation as one of the stable states of $\mathfrak{c}$. Thus, $s_{\mathbb{c}}$ is equal to one of the opposite instantiations of the positive virtual cycle $\mathfrak{c}$. □

Note that when a cycle does not contain two '$\pm$' arcs in its interaction graph, then in the set of stable states with a non-absorbing upstream instantiation, only two profiles of the cycle variables are possible. Indeed, in this case, only one virtual positive cycle $\mathfrak{c}$ can exist. Thus, the two possible profiles are the pair of opposite instantiations of $\mathfrak{c}$.

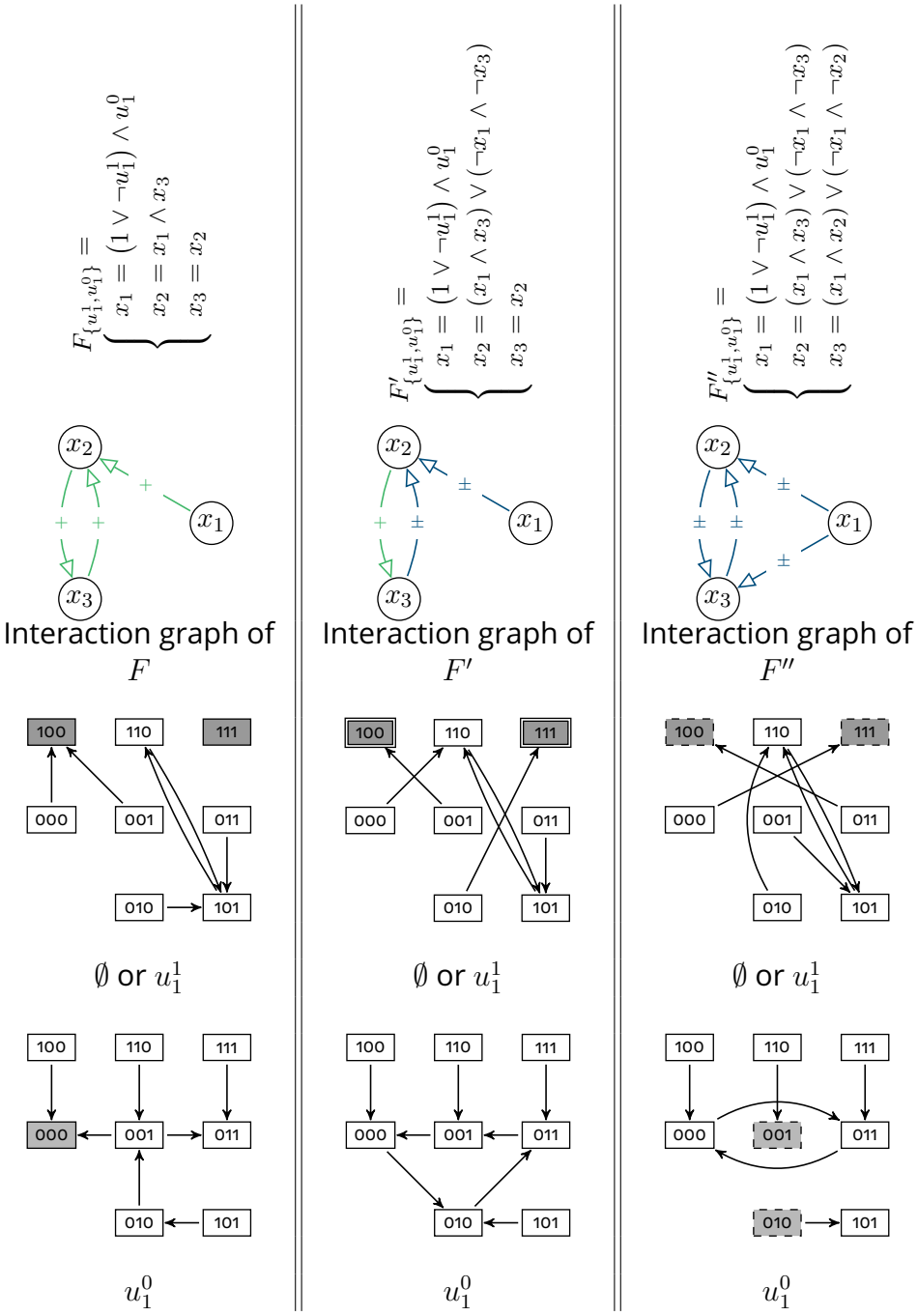
[3]Note, such a relationship is only possible when $y_j \xrightarrow{\pm} y_i$ is not the only arc in which y_i is the head of the arc (*i.e.*, there exists an arc between $x \to y_i$ with $x \in \{z_1, \ldots, z_l\}$).

When a cycle $\mathbb{c}$ does contain two or more '$\pm$' arcs in its interaction graph, $\mathbb{c}$ can have multiple pairs of opposite instantiations. When the upstream variables of $\mathbb{c}$ are in one of their non-absorbing instantiations, its set of '$\pm$' arcs becomes '$+$' or '$-$' arcs. Thus, the cycle that $\mathbb{c}$ is considered to behave as may not always be the same, because multiple positive cycles can be generated by transforming the set of '$\pm$' arcs.

To summarise, a non-negative cycle will have a set of opposite instantiations pairs. If a cycle $\mathbb{c}$ is positive or if the number of '$\pm$' arcs is fewer than 2, then the set will be of size 1. In this case, we denote for the sake of simplicity the opposite profiles as $\gamma_{\mathbb{c}}^{\top}$ and $\gamma_{\mathbb{c}}^{\bot}$.

Example 7.1. *Consider the Boolean networks F, F' and F'' of Figure 7.3. Each BCN $F_{\{u_1^1,u_1^0\}}$, $F'_{\{u_1^1,u_1^0\}}$,and $F''_{\{u_1^1,u_1^0\}}$ can freeze x_1 to 0 $(u_1^0 = 0)$ and 1 $(u_1^1 = 0)$ or not freeze x_1 $(\emptyset)$. In all these cases, the control input u_1^1 results in the same dynamics as the control input $\emptyset$. F, F', and F'' have in their $\bar{C}_X$-interaction graph a cycle $\mathbb{c} = \{x_2, x_3\}$ with, respectively, zero, one, and two '$\pm$' arcs, and the set of upstream variables of each cycle is $\mathbb{c}^{\swarrow} = \{x_1\}$. We distinguish the following situations in the three networks:*

- *$F_{\{u_1^1,u_1^0\}}$ has for a set of non-absorbing instantiations $N_{\mathbb{c}} = \{\{x_1 = 1\}\}$ and for absorbing instantiations $A_{\mathbb{c}} = \{\{x_1 = 0\}\}$. For all stable states s where $s_{\mathbb{c}^{\swarrow}} \in N_{\mathbb{c}}$, the upstream of $\mathbb{c}$ will has no impact on the dynamics of the cycle. In this case, $\mathbb{c}$ has for $\gamma_{\mathbb{c}}^{\top} = \{x_2 = 1, x_3 = 1\}$ and $\gamma_{\mathbb{c}}^{\bot} = \{x_2 = 0, x_3 = 0\}$.*
- *$F'_{\{u_1^1,u_1^0\}}$ has for set of non-absorbing instantiations $N_{\mathbb{c}} = \{\{x_1 = 1\}, \{x_1 = 0\}\}$ and for absorbing instantiations $A_{\mathbb{c}} = \emptyset$.For all stable states s where $x_1 = 0$, the cycle $\mathbb{c}$ will have the same dynamics as a negative cycle with a '$+$' arc between x_2 and x_3 and a '$-$' arc between x_3 and x_2. For all stable states s where $x_1 = 1$ the cycle $\mathbb{c}$ will have the same dynamics as the equivalent positive cycle $\mathfrak{c}$ with a '$+$' arc between x_2 and x_3 and between x_3 and x_2. The cycle $\mathfrak{c}$ has for opposite instantiations $\gamma_{\mathfrak{c}}^{\top} = \{x_2 = 1, x_3 = 1\}$ and $\gamma_{\mathfrak{c}}^{\bot} = \{x_2 = 0, x_3 = 0\}$.*
- *$F''_{\{u_1^1,u_1^0\}}$, has for a set of non-absorbing instantiations $N_{\mathbb{c}} = \{\{x_1 = 1\}, \{x_1 = 0\}\}$ and for absorbing instantiations $A_{\mathbb{c}} = \emptyset$. For all stable states s where $s_{\mathbb{c}^{\swarrow}} \in N_{\mathbb{c}}$, when $x_1 = 1$, the cycle $\mathbb{c}$ has the same dynamics as the equivalent positive cycle $\mathfrak{c}$ with a '$+$' arc between x_2 and x_3 and between x_3 and x_2. When, on the other hand, $x_1 = 0$, the cycle $\mathbb{c}$ has the same dynamics as the*



Legend: The synchronous dynamics of Boolean control networks $F_{\{u_1^1,u_1^0\}}$, $F'_{\{u_1^1,u_1^0\}}$, and $F''_{\{u_1^1,u_1^0\}}$. The dynamics are synchronous and the self-loops on states are not shown. The stable states of each dynamics are coloured in two shades of grey, and their contours are drawn in different styles. Each shade of grey is associated with different control input and each contour style is associated with a different Boolean network.

Figure 7.3: Influence of upstream instantiations on a cycle behaviour.

equivalent positive cycle $\mathfrak{c}'$ with a '$-$' arc between x_2 and x_3 and between x_3 and x_2. The cycle $\mathfrak{c}$ has for opposite instantiations $\gamma_{\mathfrak{c}}^{\top} = \{x_2 = 1, x_3 = 1\}$ and $\gamma_{\mathfrak{c}}^{\perp} = \{x_2 = 0, x_3 = 0\}$ and the cycle $\mathfrak{c}'$ has for opposite instantiations $\gamma_{\mathfrak{c}'}^{\top} = \{x_2 = 0, x_3 = 1\}$ and $\gamma_{\mathfrak{c}'}^{\perp} = \{x_2 = 1, x_3 = 0\}$.

From the definition regarding absorbing and non-absorbing instantiations, we can make two observations about the significant effect the upstream variables of a cycle have on a sequence of control.

- If the dynamics of a Boolean control network has a stable state s where $s_{\mathbb{c}}$ is not in one of the opposite instantiations of the cycle, then the upstream variables of $\mathbb{c}$ should be instantiated to one of their absorbing instantiations (*i.e.*, $s_{\mathbb{c}^{\swarrow}} \in \Gamma^{\odot}$).

- Consider a Boolean network in which none of the non-negative cycles of the $\bar{C}_X$-interaction graph are connected. In this case, all stable states s^i of a minimal sequence $\mu[k]$ under the ConEvs model of dynamics must have the upstream of at least one of the positive cycles in the $\bar{C}_X$-interaction graph set to a non-absorbing instantiation.

7.3 . Maximal sizes of minimal sequences

In Chapter 6, we presented two algorithms inferring minimal sequences of control based on the exploration of all $\bar{C}_X$-profiles. Studying the topology of the $\bar{C}_X$-interaction graph enables us to limit the number of $\bar{C}_X$-profiles to explore by providing a more accurate specification of the upper bound. By relying on the properties of the non-negative cycles presented in Section 7.2, we determine tight upper bounds on the size of control sequences.

In the first subsection, we present new upper bounds for the general case when the subgraph of $\bar{C}_X$-variables contains strongly connected components. In the second subsection, we present new upper bounds for when the subgraph of $\bar{C}_X$-variables contains a single positive cycle. Treating this particular case provides insight into the mechanism behind the necessity of sequential control. In the final subsection, we draw conclusions about the topological approach and engage a discussion on mechanisms being sequential control.

7.3.1 . Bounds in the general case

In this subsection, we treat all Boolean networks that have a $\bar{C}_X$-interaction graph containing strongly connected components. As we showed in the previous section, for a sequence to be of a size greater than or equal to two, a cycle that is not negative must at least be present in the $\bar{C}_X$-variables. Therefore, we treat networks with at least a non-negative cycle in its $\bar{C}_X$-interaction graph.

To prove the following bounds, we generalise the notion of non-absorbing instantiations, so that it can be applied to any given SCC of F, in which F is a Boolean network. Therefore, an SCC denoted $\mathcal{A}$ of F may have a set of *non-absorbing instantiations* $N_{\mathcal{A}}$ of its upstream variables in which $\mathcal{A}$ can be considered to behave as if the SCC had no upstream variables. We define the sets of variables $\{y_1, \ldots, y_k\} = \mathcal{A}$ and $\{z_1, \ldots, z_l\} = X \setminus \mathcal{A}$.

$$\begin{aligned} &\forall s \in S_X : s_{\mathcal{A}^{\swarrow}} \in N_{\mathcal{A}} \implies \\ &\forall y_i \in \mathcal{A} : \Phi(f_{y_i}(y_1, \ldots, y_k, s_{z_1}, \ldots, s_{z_l})) \notin \{1, 0\}. \end{aligned} \tag{7.5}$$

The topological approach is based on Theorem 7.3, which shows that a sequence of size greater than or equal to two can only occur in the presence of a non-negative cycle in the $\bar{C}_X$-interaction graph. This theorem and the structural properties of cycles presented in Section 7.2 enable us to prove Theorem 7.6.

Theorem 7.6 is central for understanding the causality of the sequence by highlighting the role of non-negative cycles in the sequence mechanism. From Theorem 7.4, we can extract the following observation: For a minimal sequence to be of a size greater than or equal to two, every stable state s^i of its contracted dynamics, with $2 < i$, must have at least two stable attractors. From this observation, we conclude that all s^i states must have at least one cycle in one of their opposite instantiations.

Theorem 7.6. *Let F_U be a totally controlled Boolean control network and let $\mathcal{C}$ denotes the set of non-negative cycles in its $\bar{C}_X$-interaction graph. Let $\mu^T_{[k]}$ be a minimal sequence of total controls of size $k > 2$ and resolving the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ under the ConEvs dynamics in synchronous update mode with the following trajectory: $s^1 \xrightarrow{\mu^T_1}{}^{*} s^2 \ldots s^k \xrightarrow{\mu^T_k}{}^{*} s^{k+1}$, where $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$ and s^i with $1 \leq i \leq k+1$ are stable states. In this case, all states s^{j+1} with $1 < j < k+1$ will have at least one cycle of $\mathcal{C}$ having for a profile one of its opposite instantiations.*

Proof. Since s^{i+1} is a stable state, there must exist a total control μ^T such that $F_{\mu^T}(s^{i+1}) = s^{i+1}$. Furthermore, for the state s^{i+2} not to be directly reachable from s^1, the Boolean network F_{μ^T} must have at least two attractors (Theorem 7.4). Since the control is total, all attractors of F_{μ^T} will have the same C_X profile.

We note that since s^{i+1} is stable, any negative cycle $\mathbb{c}$ of the SCCs of the $\bar{C}_X$-interaction graph must have $s^{i+1}_{\mathbb{c}^\swarrow} \in A_\mathbb{c}$. Thus, there must exist a sub-SCC $\mathcal{A}$ that has $s^{i+1}_{\mathcal{A}^\swarrow} \in N_\mathcal{A}$ and does not contain negative cycles. Indeed, if no such sub-SCC A exists,variables in all SCCs will depend on their upstream variables. Since all possible attractors have the same C_X-profile under μ_T, only one stable state attractor can exist, which is a contradiction. □

Theorem 7.6 enables us to define new upper bounds on the sequence length. We first only considered networks whose $\bar{C}_X$-interaction graph forms a single SCC, and which operate under total control. We now introduce the following notations:

- $\Gamma^{\top\perp}$ denotes the union of all pairs of opposite instantiations of a cycle.
- S_X^{stbl} denotes the set of stabilisable states $\{s \mid s \in S_X, \exists \mu \in S_U : STBL_{F_\mu}(s)\}$.
- β denotes the cardinality of the set of $\bar{C}_X$-profile in which a stabilisable state with this profile exists and in which there is a cycle in one of its opposite instantiations. More formally, $\beta = |\{s \mid s \in S_{\bar{C}_X}, \exists \mathbb{c} \in \mathcal{C}, \exists \gamma^\bullet \in \Gamma^{\top\perp}, \exists s' \in S_X^{\text{stbl}}, \exists \mu \in S_U : s_\mathbb{c} = \gamma^\bullet \wedge s'_{\bar{C}_X} = s\}|$, with $\mathcal{C}$ being the set of non-negative cycles in the $\bar{C}_X$-interaction graph.

Theorem 7.7. *Let F_U be a totally controlled Boolean control network. All minimal sequences of total controls resolving the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ under the ConEvs dynamics in synchronous update mode cannot be of a size greater than $\min(\beta + 2, 2^{|\bar{C}_X|})$.*

Proof. Take the trajectory induced by $\mu^T_{[k]} : T = s^1 \xrightarrow{\mu_1^T}{}^* s^2 \dots s^k \xrightarrow{\mu_k^T}{}^* s^{k+1}$, where $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$ and s^i, $1 < i < k+1$ are stable states.

According to Corollary 5.1, in TCS any given $\bar{C}_X$-profile cannot appear twice between s^1 and s^k. Thus, k will have for upper bound the number of different possible $\bar{C}_X$-profiles. According to Theorem 7.6, all states s^{i+1}, where $1 < i < k+1$, must have at least one cycle set to one of its opposite profiles. Thus, we can deduce that $k - 2 \leq \beta$, which proves the statement of the theorem. □

Theorem 7.7 provides a bound on the length of minimal sequences, which dependents on the number of equivalence classes in which a cycle in the $\bar{C}_X$-interaction graph is instantiated to one of its opposite instantiations. If the number of such equivalence classes for a Boolean network validating the condition of the theorem is 3, a minimal sequence will necessarily be of a size lower or equal to 5. We note that since we are in ConEvs dynamics, such an equivalence class should at least have an enduring state; otherwise, it would not be possible to stabilise a state of this equivalence class.

Note that s^1 and s^2 are the only states in which the $\bar{C}_X$-profile may contain no non-negative cycle instantiated to one of its opposite profiles. However, if a $\bar{C}_X$-variable has a self-loop with a '$+$' or '$\pm$'sign, then it will always be instantiated to one of its opposite profiles. In this case, the length of the sequence can reach the bound of $2^{|\bar{C}_X|}$ (Theorem 5.2) as $S_{\bar{C}_X} \setminus \beta = \emptyset$ (*i.e.*, the set of states in which no cycle is in one of its opposite instantiations is an empty set).

Example 5.1 of Section 5.3 is such a case. It is also, therefore, an example of a network reaching the upper bound of Theorem 7.7.

Note that it is possible to make a bound easier to calculate by ignoring the necessity of stable states:

Corollary 7.1. *Let F_U be a totally controlled Boolean control network and let $\mathcal{C}$ denotes the set of non-negative cycles in its $\bar{C}_X$-interaction graph. A control sequence resolving the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ for the ConEvs model of dynamics cannot be of a size greater than $\min(2 + |\{s \mid s \in S_{\bar{C}_X}, \exists \mathfrak{c} \in \mathcal{C}, \exists \gamma^\bullet \in \Gamma^{\top\perp} : s_\mathfrak{c} = \gamma^\bullet\}|, 2^{|\bar{C}_X|})$.*

Proof. $\beta \subseteq \{s \mid s \in S_{\bar{C}_X}, \exists \mathfrak{c} \in \mathcal{C}, \exists \gamma^\bullet \in \Gamma^{\top\perp} : s_\mathfrak{c} = \gamma^\bullet\}$,,therefore, the above statement is true. □

In the following, we extend this result to all TCS networks. We first prove that, under total control, the downstream and the disconnected variables of the $\bar{C}_X$-interaction graph have no influence on the upper bound under a TCS control strategy. We now introduce the following notations:

- $\mathcal{V}$ denotes the set of variables belonging to the SCCs of an $\bar{C}_X$-interaction graph.
- $\mathcal{V}^{\swarrow \bar{C}_X}$ denotes the set of upstream variables of $\mathcal{V}$ in the $\bar{C}_X$-interaction graph.

- $\mathcal{V}^{\diamond}$ denotes the union of $\mathcal{V}^{\swarrow \bar{C}_X}$ and $\mathcal{V}$. More formally, $\mathcal{V} \cup \mathcal{V}^{\swarrow \bar{C}_X}$.
- $\mathcal{V}^{\Downarrow}$ denotes the set of strictly downstream and disconnected variables of $\mathcal{V}$ in the $\bar{C}_X$-interaction graph. More formally, $\mathcal{V}^{\Downarrow} = \bar{C}_X \setminus \mathcal{V}^{\diamond}$.

Lemma 7.2. *Let F_U be a totally controlled Boolean control network. Let $\mu^T_{[k]}$ be any minimal sequence of total controls of size $k > 2$ that resolves the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ under the ConEvs dynamics in synchronous update mode with the following trajectory: $s^1 \xrightarrow{\mu^T_1}{}^{*} s^2 \dots s^k \xrightarrow{\mu^T_k}{}^{*} s^{k+1}$, in which s^i, where $1 \leq i \leq k+1$, are stable states. In this case, there cannot exist a pair of states s^i and s^j, where $1 \leq i < j < k+1$, which has the same $\mathcal{V}^{\diamond}$-profile.*

Proof. Assume there exist two states, s^i and s^j, in the trajectory T where $1 \leq i < j < k+1$, $s^j_{\bar{C}_X} \neq s^l_{\bar{C}_X}$ and $s^i_{\mathcal{V}^{\diamond}} = s^j_{\mathcal{V}^{\diamond}}$. By definition, all paths from a $\mathcal{V}^{\Downarrow}$ variable to a variable in $\mathcal{V}$ must contain some C_X-variables. Thus, since all C_X-variables are totally controlled, the profile of $\mathcal{V}^{\Downarrow}$-variables cannot influence the profiles of $\mathcal{V}$ variables. Therefore, the profile $s^{j+1}_{\bar{C}_X}$ only dependents on $s^j_{\mathcal{V}^{\diamond}}$. In this case, s^{j+1} can be reached from s^i under the total control μ^T_j. Thereby, we conclude that there cannot be two states in the trajectory of a minimal sequence of control total that have the same $\mathcal{V}^{\diamond}$ profile. □

We then give an upper bound on the sequence length as a function of the number of equivalence classes whose states have a cycle in the $\bar{C}_X$-interaction graph instantiated to one of its opposite instantiations. Note that if we obtain two equivalence classes that only differ in the downstream or disconnected variables, then visiting only one of them is sufficient. We now introduce the following notations:

- β' denotes the cardinality of the set of $\mathcal{V}^{\diamond}$-profile in which a stabilisable state with this profile exists and in which there is a cycle in one of its opposite instantiations. More formally, $\beta' = |\{s \mid s \in S_{\mathcal{V}^{\diamond}}, \exists \mathbb{c} \in \mathcal{C}, \exists \gamma^{\bullet} \in \Gamma^{\top\perp}, \exists s' \in S^{\mathsf{stbl}}_X : s_{\mathbb{c}} = \gamma^{\bullet} \wedge s'_{\bar{C}_X} = s\}|$

Theorem 7.8. *Let F_U be a totally controlled Boolean control network. All minimal sequences of total controls resolving the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ under the ConEvs dynamics in synchronous update mode cannot be of a size greater than $\min(\beta' + 2, 2^{|\bar{C}_X|})$.*

Proof. Take the trajectory induced by $\mu^T_{[k]} : T = s^1 \xrightarrow{\mu^T_1}{}^* s^2 \dots s^k \xrightarrow{\mu^T_k}{}^* s^{k+1}$, where $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$ and s^i, $1 < i < k+1$ are stable states.

We know from Theorem 7.6 that all states s^{i+1} with $1 < i < k+1$ must have at least one of their cycles set at one of it opposite instantiation. Futhermore, according to Lemma 7.2, no a pair of states s^i and s^j , where $1 \leq i < j < k+1$, and that have the same $\mathcal{V}^\diamond$ exist in the trajectory. Thus, we can deduce that $k-2 \leq \beta'$., which proves the statement of the theorem. □

Regarding Theorem 7.6, Example 5.1 of Section 5.3 is an example of a network reaching the upper bound of Theorem 7.8

We now further generalise this result from TCS to OCS control sequences. Lemma 7.3 proves that the downstreams of the SCCs and the disconnected variables have no influence on the lengths of minimal OCS control sequences.

Lemma 7.3. *Let F_U be a Boolean control network. Let $\mu_{[k]}$ be a minimal sequence of control of size $k > 2$ that resolves the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ under the ConEvs dynamics in synchronous update mode with the following trajectory $s^1 \xrightarrow{\mu_1}{}^* s^2 \dots s^k \xrightarrow{\mu_k}{}^* s^{k+1}$, where s^i, $1 \leq i \leq k+1$, are stable states. In this case, there exists at most a pair of states in the trajectory that has the same $\mathcal{V}^\diamond$-profile.*

Proof. Assume there exist three states, s^i, s^j and s^l, in the trajectory T where $1 \leq i < j < l \leq k+1$ and the following conditions hold:

- $s^i_{\bar{C}_X} \neq s^j_{\bar{C}_X} \wedge s^i_{\bar{C}_X} \neq s^l_{\bar{C}_X} \wedge s^j_{\bar{C}_X} \neq s^l_{\bar{C}_X}$.
- $s^i_{\mathcal{V}^\diamond} = s^j_{\mathcal{V}^\diamond} = s^l_{\mathcal{V}^\diamond}$.

Here, the only differences between the three states are the values of downstream and disconnected variables of $\mathcal{V}$ in the $\bar{C}_X$-interaction graph (*i.e.*, $\mathcal{V}^\Downarrow$).

Since s^l is a stable state, there must exist a total control μ^T such that $F_{\mu^T}(s^l) = s^l$. By definition, all paths from a $\mathcal{V}^\Downarrow$ variable to a $\mathcal{V}$ variable must contain some C_X-variables. Thus, in the case of the total control μ^T, $\mathcal{V}^\Downarrow$ variables cannot influence the profiles of $\mathcal{V}$ variables. Thus, s^l must be reachable under μ^T from any enduring state s with the property $s^l_{\mathcal{V}^\diamond} = s_{\mathcal{V}^\diamond}$. Therefore, s^l can be reached from s^i. Thereby, the sequence $\mu_{[k]}$ cannot be minimal, which is a contradiction. □

Theorem 7.9. *Let F_U be a Boolean control network. All minimal sequences of control $\mu_{[k]}$ resolving the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ under the ConEvs dynamics in synchronous update mode will be of size $\min(2\beta' + 2, 2^{|\mathcal{V}^\diamond|+1})$.*

Proof. Take the trajectory induced by $\mu_{[k]} : T = s^1 \xrightarrow{\mu_1}{}^* s^2 \dots s^k \xrightarrow{\mu_k}{}^* s^{k+1}$, where $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$, and s^i, where $1 < i < k+1$, are stable states.

We know from Theorem 7.6 that all states s^{i+1} with $1 < i < k+1$ must have at least one of their cycles set at one of its opposite instantiations. Furthermore, according to Lemma 7.2 there cannot exist three states in the trajectory that have the same $\mathcal{V}^\diamond$-profile, which proves the statement of the theorem. □

Theorem 7.9 is similar to Theorem 7.8 but considers the possibility of having duplicates (Section 5.3). With duplicates, the upper bound becomes $2^{|\bar{C}_X|+1}$ if the SCCs do not have downstream variables (*i.e.*, $\bar{C}_X = \mathcal{V}^\diamond$). Thus, Example 5.2 of Section 5.3 is an example of a network reaching the upper bound.

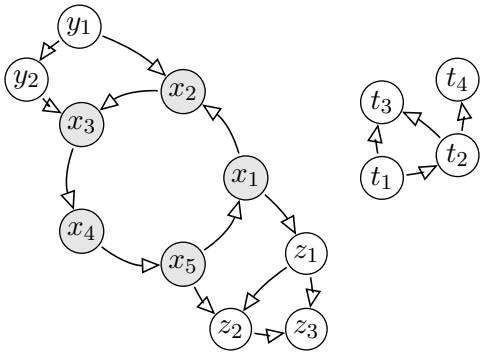
7.3.2 . Bounds for a single positive cycle

In this subsection, we examine at the networks in which the $\bar{C}_X$-interaction graph consists of a unique positive cycle. This interaction graph is possibly equipped with acyclic appendages and accompanied by disconnected acyclic structures, as in Figure 7.4. Such configurations may lead to minimal sequences of length two at most, as shown in the following theorem.

Theorem 7.10. *Let F_U be a totally controlled Boolean control network whose $\bar{C}_X$-interaction graph consists of a unique positive cycle. In this case, all minimal sequences of total control $\mu^T_{[k]}$ resolving the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ under the ConEvs dynamics in synchronous update mode are of size ≤ 2.*

Proof. Take the trajectory induced by $\mu^T_{[k]} : T = s^1 \xrightarrow{\mu^T_1}{}^* s^2 \dots s^k \xrightarrow{\mu^T_k}{}^* s^{k+1}$, where $s^1 \in S_\alpha$, $s^{k+1} \in S_\omega$ and s^i, where $1 < i < k+1$, are stable states. We denote $\mathbb{c}$ as the unique cycle of the $\bar{C}_X$-interaction graph.

We know from Theorem 7.6 that for all states s^{j+1} with $1 < j < k+1$, at least one cycle in the $\bar{C}_X$-interaction graph is in one of its opposite instantiations. Moreover, to reach s^{j+1} from s^j, s^j and s^{j+1} must be in the same basin of attraction in the dynamics of $F_{\mu^T_j}$.



Legend: The $\bar{C}_X$-variables x_1 to x_5 form a positive cycle. The $\bar{C}_X$-variables y_1 and y_2 form its *upstream*, the $\bar{C}_X$-variables z_1 to z_3 form its *downstream*, while the $\bar{C}_X$-variables t_1 to t_4 form an acyclic structure disconnected from the positive cycle. Only the cycle is required to be positive. The signs on all other arcs in the graph can be arbitrary. The connections to controlled variables are not shown.

Figure 7.4: The single positive cycle on $\bar{C}_X$, accompanied by upstream, downstream, and disconnected acyclic structures.

Since the $\bar{C}_X$-interaction graph consists of a unique positive cycle, all $s^{j+1}_{\mathbb{c}^\swarrow}$ are in $N_{\mathbb{c}}$, meaning the variables of $\mathbb{c}$ have the same dynamics as the cycle $\mathbb{c}$ without upstream variables. Since $\mathbb{c}$ has a unique positive virtual cycle, we know from the first remark of Proposition 7.1 that the cycle $\mathbb{c}$ only has a single pair of opposite instantiations $\{\gamma^\top_{\mathbb{c}}, \gamma^\bot_{\mathbb{c}}\}$. Thus, all state s^{j+1} have the same basins of attraction. Therefore, to reach s^{j+1} from a state s^j where $s^{j+1}_{\mathbb{c}} \neq s^j_{\mathbb{c}}$, μ^T_j must be a control generating a network with only one attractor (*i.e.*, $j = 1$). Moreover, from s^{j+1}it is possible to reach all states s in which $s^{j+1}_{\mathbb{c}} = s_{\mathbb{c}}$ in one-step since $s$and s^{j+1} are two enduring states of the same equivalence class (Proposition 5.2). Therefore, a minimal sequence has a maximal size of 2. □

This result can be generalised from TCS to the general OCS case. We begin by revealing that in Lemma 7.4 all non-total controls can always be replaced by a total control. This process enables us to prove the following theorem, which provides the same upper bound on the lengths of minimal OCS sequences as in Theorem 7.10.

Lemma 7.4. *Let F_U be a controlled Boolean control network. Assume that its $\bar{C}_X$-interaction graph consists of a unique positive cycle $\mathbb{c}$ with $\{\gamma^\top_{\mathbb{c}}, \gamma^\bot_{\mathbb{c}}\}$ as its unique pair of opposite instantiations. Let s and s' be enduring states, with $s'_{\mathbb{c}} \in$*

$\{\gamma_{\mathbb{c}}^{\top}, \gamma_{\mathbb{c}}^{\perp}\}$ *and* S_{U^T} *the set of total controls. The following statement is true under the synchronous update mode:*

$$(\exists \mu \in S_U : s \xrightarrow{\mu}{}^{*} s' \wedge STBL_{F_\mu}(s')) \implies$$
$$(\exists s'' \in S_X,\ \exists \mu^T \in S_{U^T} : s \xrightarrow{\mu^T}{}^{*} s'' \wedge STBL_{F_{\mu^T}}(s'') \wedge s'_{\mathbb{c}} = s''_{\mathbb{c}}).$$

Proof. Assume there is a control μ such that $s \xrightarrow{\mu}{}^{*} s' \wedge STBL_{F_\mu}(s')$ but there is no total control μ^T such that:

$$s \xrightarrow{\mu^T}{}^{*} s' \wedge STBL_{F_{\mu^T}}(s'') \wedge s'_{\mathbb{c}} = s''_{\mathbb{c}}. \tag{7.6}$$

Here $s'_{\mathbb{c}^{\swarrow}}$ must be a non-absorbing instantiation. If this were not the case, it would suffice to apply the total control, which would put the upstream in the configuration of $s'_{\mathbb{c}^{\swarrow}}$ to reach s'. Moreover, $s_{\mathbb{c}} \neq s'_{\mathbb{c}}$ since if $s_{\mathbb{c}}$ was equal to $s'_{\mathbb{c}}$, it would suffice to apply the total control which would put the upstream in the configuration of $s'_{\mathbb{c}^{\swarrow}}$ to reach s' as s and s' are two enduring states of the same equivalence class (Proposition 5.2).

Since $s'_{\mathbb{c}^{\swarrow}} \in N_{\mathbb{c}}$ and $s'_{\mathbb{c}} \neq s'_{\mathbb{c}}$, there must be a transition $s^i \xrightarrow{\mu} s^{i+1}$ in the uncontracted trajectory $s \xrightarrow{\mu} \ldots \xrightarrow{\mu} s'$ such that $s^{i+1}_{\mathbb{c}} = s'_{\mathbb{c}}$ and $s^i_{\mathbb{c}} \neq s'_{\mathbb{c}}$.

If $s^i_{\mathbb{c}^{\swarrow}} \in N_{\mathbb{c}}$, then the instantiation $s^i_{\mathbb{c}}$ must be part of a cyclic attractor of the virtual positive cycle $\mathfrak{c}$ of the cycle $\mathbb{c}$ since s^i is not a stable state and all states of $\mathfrak{c}$ belong to an attractor [18]. Thus, $s^{i+1}_{\mathbb{c}} = s'_{\mathbb{c}}$ cannot be true as $s'_{\mathbb{c}}$ is one of the stable states of $\mathfrak{c}$. Therefore, $s^i_{\mathbb{c}^{\swarrow}}$ cannot be in a non-absorbing instantiation. This means that $s^i_{\mathbb{c}^{\swarrow}} \in A_{\mathbb{c}}$.

Let μ^T be a total control that freezes the C_X-variables at the instantiation of the C_X-profile of s^i. Such control leads to the following trajectory $s^i \xrightarrow{\mu^T} s^{i+1\prime} \xrightarrow{\mu^T} s^{i+2\prime}$ in which $s^{i+1\prime}_{\mathbb{c}} \in \{\gamma_{\mathbb{c}}^{\top}, \gamma_{\mathbb{c}}^{\perp}\}$ since $s^{i+1\prime}_{\mathbb{c}} = s^{i+1}_{\mathbb{c}} = s'_{\mathbb{c}}$.

If $s^{i+1\prime}_{\mathbb{c}} = s^{i+2\prime}_{\mathbb{c}}$, then $s^{i+1\prime}$ is a stable state as all $\mathbb{c}^{\swarrow}$-variables under the total control μ^T are C_X-variables. Since $s^i_{\mathbb{c}^{\swarrow}} \in A_{\mathbb{c}}$, the controlled Boolean network F_{μ^T} would have only one stable state with no cyclic attractors. Therefore, Equation 7.6 would be true.

Suppose now that $s^{i+1\prime}_c \neq s^{i+2\prime}_c$. Since the total control μ^T fixes the C_X-profiles of s^i and its subsequent states to the same constants, the change of instantiation of $\bar{C}_X$-variables depends on their formula. Thus, the dynamics generated by the formula of the variables of the cycle $\mathbb{c}$ must contains the

trajectory $s^i \xrightarrow{\mu^T} s^{i+1'} \xrightarrow{\mu^T} s^{i+2'}$, in which $s^i_{\mathbb{c}} \neq s^{i+1'}_{\mathbb{c}} \wedge s^{i+1'}_{\mathbb{c}} \neq s^{i+2'}_{\mathbb{c}}$ and $s^{i+1'}_{\mathbb{c}} \in \{\gamma^{\top}_{\mathbb{c}}, \gamma^{\perp}_{\mathbb{c}}\}$. Such a trajectory cannot be generated by a positive cycle with its upstream variables in absorbing or non-absorbing instantiation., which proves the statement of the theorem.

□

Theorem 7.11. *Let F_U be a totally controlled Boolean control network. Assume that its $\bar{C}_X$-interaction graph consists of a unique positive cycle. In this case, all minimal sequences of control resolving the CoFaSe problem $(F_U, S_\alpha, S_\omega)$ under the ConEvs dynamics in synchronous update mode are of size ≤ 2.*

Proof. Since the $\bar{C}_X$-interaction graph is a positive cycle, for all controls reaching a stable state s' from s, there is a total control reaching the stable state s' from s (Lemma 7.4). Thereby, we conclude that if a sequence validates the CoFaSe problem under the ConEvs dynamics in synchronous update mode, a sequence of total controls validating the same problem also exists. The upper bound, is thus, the same as in Theorem 7.10, which proves the statement of the theorem. □

We insist on the fact that Theorems 7.10 and 7.11 are only applied for positive cycles and *not* positive/negative cycles. For example, the Boolean network from Figure 3.1 in Section 3.1, which has for $\bar{C}_X$-interaction graph a unique positive/negative cycle $\{x_1\}$, one can find minimal sequences of size 3 (Trajectory 3.7).

From the two theorems, we understand that a minimal sequence of control sets different cycles to profiles, which will drive them into one of their opposite instantiations in a specific order.

Example 7.2. *The controlled Boolean network $F_{\{u_1^0, u_1^1\}}$ of Figure 7.2, with $\bar{C}_X = \{x_2\}$, is a prime example of a Boolean network reaching the bounds of Theorems 7.10 and 7.11. For the CoFaSe problem under the ConEvs dynamics, in which $S_\alpha = \{11\}$ and $S_\omega = \{01\}$, the solution for the TCS and OCS control strategies is the following sequence of size 2:*

$$11 \xrightarrow{\{u_1^0\}}{}^{*} 00 \xrightarrow{\{u_1^1\}}{}^{*} 01$$

To facilitate reading, we repeat the controlled Boolean network equations below:

$$F_{\{u_1^1, u_1^0\}} = \begin{cases} x_1 = (x_1 \vee \neg u_1^1) \wedge u_1^0 \\ x_2 = x_1 \wedge x_2 \end{cases}$$

In Example 7.2, the attractor we want to reach is when the cycle $\mathbb{c} = \{x_2\}$ has x_1 set to 0. In S_α, x_2 is set to 1. Since the state 01 can be stabilised only by networks $F_{\{u_1^1\}}$ and $F_\emptyset$, in which 11 is not in the desired attractor, we must drive the controlled network in a intermediary step to a state that would eventually end up in the desired attractor. This is realised by setting x_2 to 0.

7.3.3 . Discussion

This topological approach enables us to determine tight upper bounds. The approach limits the number of $\bar{C}_X$-profiles to explore and thus decreases the computation time. The most notable reduction is due to Lemmas 7.2 and 7.3, which reveal that the downstream and disconnected variables of the SCCs have no influence on the lengths of minimal sequences. Thus, a sequence can be found by considering the equivalence classes of $\mathcal{V}^\diamond$-variables.

On the other hand, knowing that every[4] stable state of a sequence dynamics must have at least one cycle in one of its opposite instantiations does not create any significant gain in algorithm computation time. Indeed, the stabilisable states in which no cycle is in its opposite instantiations can be reached from any states with a total control. Thus, they would be all reached from S_α at the first iteration of the algorithms presented in Chapter 6. However, sequences that traverse such states in their trajectory are uncommon as they require complex Boolean formulas. Therefore, trying at first to infer sequences of control without considering states in which no cycle is in its opposite instantiations can be a good method for obtaining a better computation time in general.

From Theorems 7.10 and 7.11,we can extract some interesting observations concerning a BCN whose $\bar{C}_X$-interaction graph contains a single SCC, which is a cycle. For example, if a sequence is of size two, the property that we wish to reach must require, according to Theorem 7.6, the $\bar{C}_X$-variables to be in one of the two opposite instantiations of the cycle. In this case, from the initial states, we have only one $\bar{C}_X$-profile to test: the profile in which the cycle is set in the same instantiation as one of the target states. In this situation, the cycle acts as a binary memory. The target states are in the 'good' memory state which can be translated into being blocked in a 'good' attractor. The initial states are in the 'bad' memory state or blocked in a 'bad' attractor. Thus, to reach the desired states, we must flip the memory to a 'good' state (*i.e.*, drive the network to a state in the right attractor).

[4] Excluding the first two stable states of the trajectory.

Under the simple premise of having for $\bar{C}_X$-interaction graph a unique positive cycle, finding the order in which cycles need to be set to 'good' attractors is trivial. In the general case, finding such an order cannot be done achieved using a topological analysis. The problem arises from the fact that the $\bar{C}_X$-interaction graph abstraction of the interactions between $\bar{C}_X$-variables is not granular enough to deduce such an order in complex topologies.

For example, if the cycle has upstream variables in it $\bar{C}_X$-variables, we must to refer to Theorem 7.8 for the TCS control strategy and Theorem 7.9 for the OCS control strategy. The example, below, shows a controlled network reaching the upper bound with a TCS control strategy and having only the positive cycle $\{x_1, x_2\}$ in its $\bar{C}_X$-interaction graph.

Example 7.3. *The Boolean network in Figure 7.5 has for $\bar{C}_X$-variables x_5, x_6 and x_7. Its $\bar{C}_X$-interaction graph contains only the positive cycle $\mathbb{c} = \{x_1, x_2\}$. It is one of the simplest examples of a Boolean network with a sequence of control reaching the bound of Theorem 7.8. In this network, there are only 8 equivalence classes in which the positive cycle $\mathbb{c}$ is in one of its opposite instantiations:*

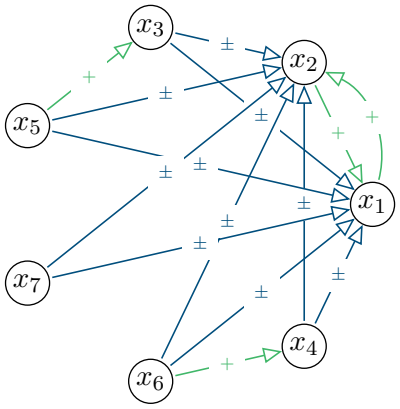
$$[0000 \star\star\star] \quad [0001 \star\star\star] \quad [0010 \star\star\star] \quad [0011 \star\star\star]$$

$$[1100 \star\star\star] \quad [1101 \star\star\star] \quad [1110 \star\star\star] \quad [1111 \star\star\star]$$

Therefore, the maximal theoretical sequence size for TCS should be $2 + 8$. This bound is reached for the following CoFaSe problem under the ConEvs dynamics where $S_\alpha = \{1001011\}$, $S_\omega = \{1111111\}$. This problem has for a solution the sequence of size 10:

$$\begin{aligned}
&1001011 \xrightarrow{\{u_5^1,u_6^0,u_7^1\}}{}^{*} 0110101 \xrightarrow{\{u_5^0,u_6^0,u_7^1\}}{}^{*} 0000001 \xrightarrow{\{u_5^0,u_6^1,u_7^0\}}{}^{*}\\
&0001010 \xrightarrow{\{u_5^1,u_6^0,u_7^0\}}{}^{*} 0010100 \xrightarrow{\{u_5^1,u_6^1,u_7^0\}}{}^{*} 0011110 \xrightarrow{\{u_5^0,u_6^0,u_7^0\}}{}^{*}\\
&1100000 \xrightarrow{\{u_5^0,u_6^1,u_7^0\}}{}^{*} 1101010 \xrightarrow{\{u_5^1,u_6^0,u_7^0\}}{}^{*} 1110100 \xrightarrow{\{u_5^1,u_6^1,u_7^0\}}{}^{*}\\
&1111110 \xrightarrow{\{u_5^1,u_6^1,u_7^1\}}{}^{*} 1111111.
\end{aligned}$$

$$
\begin{aligned}
&F_{\{u_5^1,u_5^0,u_6^1,u_6^0,u_7^1,u_7^0\}} = \\
&\begin{cases}
x_1 = (x_2 \wedge \neg x_3 \wedge \neg x_5) \vee (x_2 \wedge x_6) \vee \\
(x_2 \wedge \neg x_7) \vee (x_3 \wedge x_4 \wedge \neg x_6 \wedge \neg x_7) \vee \\
(x_3 \wedge \neg x_5 \wedge \neg x_7) \vee (\neg x_3 \wedge \neg x_4 \wedge x_5 \wedge \neg x_7) \vee \\
(\neg x_3 \wedge x_5 \wedge x_6) \vee (x_4 \wedge \neg x_5 \wedge \neg x_6) \vee \\
(\neg x_4 \wedge x_6 \wedge x_7) \vee (\neg x_5 \wedge x_6 \wedge x_7) \\
x_2 = (x_1 \wedge x_3 \wedge x_4 \wedge x_5 \wedge x_6) \vee \\
(x_1 \wedge x_3 \wedge \neg x_4 \wedge x_5 \wedge \neg x_7) \vee \\
(x_1 \wedge \neg x_3 \wedge x_4 \wedge x_5 \wedge \neg x_6) \vee \\
(x_1 \wedge \neg x_3 \wedge \neg x_4 \wedge \neg x_5 \wedge \neg x_6) \vee \\
(x_1 \wedge \neg x_3 \wedge \neg x_5 \wedge x_6 \wedge \neg x_7) \vee \\
(x_3 \wedge x_4 \wedge \neg x_5 \wedge \neg x_6 \wedge \neg x_7) \vee \\
(x_5 \wedge \neg x_6 \wedge x_7) \\
x_3 = x_5 \\
x_4 = x_6 \\
x_5 = (0 \vee \neg u_5^1) \wedge u_5^0 \\
x_6 = (1 \vee \neg u_6^1) \wedge u_6^0 \\
x_7 = (1 \vee \neg u_7^1) \wedge u_7^0
\end{cases}
\end{aligned}
$$



Legend: Boolean control network $F_{\{u_5^1,u_5^0,u_6^1,u_6^0,u_7^1,u_7^0\}}$ having in its $\bar{C}_X$-interaction graph a unique positive cycle, for which there exists a minimal control sequence reaching the bound defined in Theorem 7.8. The Boolean control network is accompanied by the interaction graph of the corresponding Boolean network without control parameters.

Figure 7.5: A Boolean network with a control sequence reaching the bound for TCS sequences .

8 - Conclusion

In this thesis, we studied the sequential control applied to Boolean networks. The results are concretely applied to network medicine, which uses the concept of networks as a modelling framework.

We proposed a new framework that defines sequential control for Boolean control networks. In particular, we considered freeze controls, under which the variables can be frozen to 0 or 1. The proposed controlled dynamics extends the Boolean network dynamics by revealing how the system evolves through a sequence of control inputs. We defined a model of controlled dynamics denoted *ConEvs*, in which the modification of the control only occurs at a stable state in the synchronous update mode.

To allow detailed theoretical analysis, the variables of the studied network were partitioned into two sets. The set of variables that are controllable were denoted C_X-variables, and the set of variables that cannot be controlled were denoted $\bar{C}_X$-variables.

We referred to the inference problem of finding a control sequence modifying the dynamics to evolve towards a desired state or property *CoFaSe* (i.e., Controlled Fate in Sequence). We proved that this problem is PSPACE-hard.

We know from the complexity of CoFaSe that finding a minimal control sequence by exhaustively exploring all possible control sequences is intractable. By seeking for factors that limit the set of the possible intermediary states of the sequence, we significantly reduced the search space in practice.

By studying the dynamical properties of the CoFaSe problem, we found that the dynamical properties that imply the necessity of a sequence of control emerged from the update functions of $\bar{C}_X$-variables. These properties enabled us to define the upper bounds on the length of minimal control sequences. We found that a minimal control sequence cannot be larger than $2^{|\bar{C}_X|+1} - 1$. These upper bounds indicate that the uncontrollable variables are central for the inference of a control sequence. As the number of uncontrolled variables is, in practice, markedly lower than the number of controlled variables (*e.g.*, [7, 11, 13]), the exhaustive exploration of all possible profiles for these variables constitutes an efficient approach for control sequence computation.

Our analysis also emphasised non-obvious complex features of the sequence, such as the occurrence of duplicates in which only the controlled variables evolve without changing the states of uncontrolled variables. Such occurrences were interpreted as the need to evolve to different stable states with the same profiles for uncontrolled variables, but which could not previously be reached.

We proposed two approaches for inferring control sequences. Both approaches are based on the exhaustive exploration of the possible intermediate profiles of uncontrolled variables [40, 41]. The first approach is implemented in Algorithm 1 (page 63) and applies the single control inference algorithm from [8] $2^{|\bar{C}_X|+1} - 1$ times in the worst case.

The second approach is implemented as a chaining of Algorithms 2 and 3 (pages 70 and 71). This approach focuses on inferring a *total* control sequence in which all controllable variables are frozen, and then reducing the sizes of individual controls. This technique implies that the second approach does not consider the occurrence of duplicates. Our benchmark revealed that inferring total controls is much cheaper because the dynamics of uncontrolled variables are neglected, while the quality of the solutions remains close to the quality of the sequences inferred by Algorithm 1.

We explored the relationship between the structure of the BCN and its dynamics to understand how the structure impacts the control sequences and thereby improves the bounds. Therefore, we studied the dynamics properties emanating from the interaction graph to characterise the bounds on the sequence length with respect to its topology. Taking topology into account makes it possible to find tighter bounds for minimal sequences of control. These upper bounds also enabled us to study the causal relationships that exist between structure and control.

One of our major observations is the importance of non-negative cycles in the interaction subgraph generated by the $\bar{C}_X$-variables ($\bar{C}_X$-*interaction graph*). We found that without a non-negative cycle in the $\bar{C}_X$-interaction graph, all properties reachable with control or a sequence of controls are reachable with a one-step control. This finding is due to the fact that for some instantiations of the upstream of a non-negative cycle, this cycle behaves as if it were an isolated subgraph, which therefore has only two opposite stable instantiations. Informally, a sequence of control is needed when an uncontrollable non-negative cycle acts as a memory that blocks the validation of the desired property until this memory is set to its un-blocking value.

We also revealed that all stable states encountered in a minimal sequence

trajectory under the ConEvs dynamics and that cannot be reached from the set of initial states must set at least one of the cycles in one of its opposite instantiations. This process results in a bound dependent on the number of $\bar{C}_X$-profiles in which at least one cycle in the $\bar{C}_X$-interaction graph is instantiated to one of its opposite instantiations.

Perspectives. The perspectives of our work are threefold:

The analysis of the Boolean functions: The proposed dynamics analysis remains general but can be improved by considering the topological aspects or the structure of formulas. In this thesis, we chose to study the topological approach. Under this approach, the inference of a sequence of control is trivial if the $\bar{C}_X$-interaction graph consists of a unique positive cycle. However simply adding some more variables and more cycles in the $\bar{C}_X$-variables rapidly complicates the situation. This is due to the fact that multiple Boolean networks can have the same topology. For more complex networks, one of the possible approaches to consider is combining the study of the topology and the analysis of the update functions of the $\bar{C}_X$-variables. Indeed, by analysing the Boolean formulas of the non-negative cycles in the $\bar{C}_X$-interaction graph, we hope to find the order in which the cycles need to be blocked or unblocked to reach a state validating the desired property. Such a discovery should greatly improve inference times, enabling the inference on larger networks.

Sequential control in biological models: This thesis focused on the formal aspects of sequential control. Applying this method to biological cases to investigate complex treatment schemes, more specifically for cancer, seems a natural prospect. One other interesting perspective would involve conducting a survey on the existing biological Boolean models to determine how many contain non-negative cycles in their set of uncontrollable variables and require a sequence of control to reach the target property from the given initial states. Such a survey could provide a good understanding of the frequency of sequential control in biological models. the findings would open a discussion on whether such frequency represented or under-represented biological mechanisms or whether it is an undesirable modelling effect.

Asynchronous CoFaSe problem: The final perspective revolves around studying sequential controls in a broader context, notably by considering other modes, such as the asynchronous one. For the asynchronous mode, one needs to tackle the fact

that the transition becomes a relation inducing non-determinism that should not be exhaustively explored to ensure the efficiency of a minimal sequential control inferring algorithm. In the synchronous algorithms presented in this manuscript, the desired property is reached from at least one of the initial states. However, because of the non-deterministic nature of the asynchronous mode, the notion of necessity and the possibility of reaching the desired property must also be introduced.

Bibliography

[1] Réka Albert. Scale-free networks in cell biology. *Journal of Cell Science*, 118(21):4947–4957, 2005.

[2] Réka Albert and Hans G Othmer. The topology of the regulatory interactions predicts the expression pattern of the segment polarity genes in drosophila melanogaster. *Journal of theoretical biology*, 223(1):1–18, 2003.

[3] Julio Aracena. Maximum number of fixed points in regulatory boolean networks. *Bulletin of mathematical biology*, 70(5):1398, 2008.

[4] Julio Aracena, Jacques Demongeot, and Eric Goles. Positive and negative circuits in discrete neural networks. *IEEE Transactions on Neural Networks*, 15(1):77–83, 2004.

[5] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[6] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nature reviews. Genetics*, 12:56–68, 2011.

[7] Célia Biane and Franck Delaplace. Abduction based drug target discovery using Boolean control network. In *Proceedings Computational Methods in Systems Biology - 15th International Conference, CMSB 2017, Darmstadt, Germany, September 27-29*, pages 57–73, 2017.

[8] Célia Biane and Franck Delaplace. Causal reasoning on Boolean control networks based on abduction: theory and application to cancer drug discovery. *IEEE/ACM transactions on computational biology and bioinformatics*, 2018.

[9] Célia Biane, Franck Delaplace, and Tarek Melliti. Abductive network action inference for targeted therapy. In *Electronic Notes in Theoretical Computer Science*, volume 335, pages 3–25, 2016.

[10] Stefan Bornholdt. Boolean network models of cellular regulation: prospects and limitations. *Journal of the Royal Society Interface*, 5(suppl_1):S85–S94, 2008.

[11] Laura N. Burga, Hai Hu, Ashish Juvekar, Nadine M. Tung, Susan L. Troyan, Erin W. Hofstatter, and Gerburg M. Wulf. Loss of BRCA1 leads to an increase in epidermal growth factor receptor expression in mammary epithelial cells, and epidermal growth factor receptor inhibition prevents estrogen receptor-negative cancers in BRCA1-mutant mice. *Breast Cancer Research*, 13(2):R30, 2011.

[12] Thomas Chatain, Stefan Haar, and Loïc Paulevé. Boolean networks: beyond generalized asynchronicity. In *International Workshop on Cellular Automata and Discrete Complex Systems*, pages 29–42. Springer, 2018.

[13] Cindy H. Chau, Olivier Rixe, Howard McLeod, and William D. Figg. Validation of analytic methods for biomarkers used in drug development. *Clinical Cancer Research*, 14(19):5967–5976, 2008.

[14] National Research Council et al. Toward precision medicine: building a knowledge network for biomedical research and a new taxonomy of disease. 2011.

[15] Pau Creixell, Erwin M. Schoof, Craig D. Simpson, James Longden, Chad J. Miller, Hua Jane Lou, Lara Perryman, Thomas R. Cox, Nevena Zivanovic, Antonio Palmeri, Agata Wesolowska-Andersen, Manuela Helmer-Citterich, Jesper Ferkinghoff-Borg, Hiroaki Itamochi, Bernd Bodenmiller, Janine T. Erler, Benjamin E. Turk, and Rune Linding. Kinome-wide decoding of network-attacking mutations rewiring cancer signaling. *Cell*, 163(1):202–217, 2015.

[16] Peter Csermely, Tamàs Korcsmàros, Huba J. M. Kiss, Gàbor London, and Ruth Nussinov. Structure and dynamics of molecular networks: A novel paradigm of drug discovery: A comprehensive review. *Pharmacology and Therapeutics*, 138(3):333–408, 2013.

[17] Jacques Demongeot, Eric Goles, Michel Morvan, Mathilde Noual, and Sylvain Sené. Attraction basins as gauges of the robustness against boundary conditions in biological complex systems. *PLoS One*, 5(8):e11793, 2010.

[18] Jacques Demongeot, Mathilde Noual, and Sylvain Sené. Combinatorics of Boolean automata circuits dynamics. *Discrete Applied Mathematics*, 160(4-5):398–415, March 2012.

[19] Jacques Demongeot and Sylvain Sené. Phase transitions in stochastic non-linear threshold Boolean automata networks on Z^2: the boundary impact. *Advances in Applied Mathematics*, 98:77–99, 2018.

[20] Jacques Demongeot and Sylvain Sené. About block-parallel boolean networks: a position paper. *Natural Computing*, 19(1):5–13, 2020.

[21] Alberto Dennunzio, Enrico Formenti, Luca Manzoni, and Antonio E. Porreca. Complexity of the dynamics of reaction systems. *Information and Computation*, 267:96–109, 2019.

[22] Eric R. Fearon and Bert Vogelstein. A genetic model for colorectal tumorigenesis. *Cell*, 61(5):759–767, 1990.

[23] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[24] Eric Goles, Fabiola Lobos, Gonzalo A Ruz, and Sylvain Sené. Attractor landscapes in boolean networks with firing memory: a theoretical study applied to genetic networks. *Natural Computing*, 19(2):295–319, 2020.

[25] Eric Goles, Pedro Montealegre, and Martín Ríos-Wilson. On the effects of firing memory in the dynamics of conjunctive networks. In *International Workshop on Cellular Automata and Discrete Complex Systems*, pages 1–19. Springer, 2019.

[26] Eric Goles and Mathilde Noual. Block-sequential update schedules and boolean automata circuits. In *Discrete Mathematics and Theoretical Computer Science*, pages 41–50. Discrete Mathematics and Theoretical Computer Science, 2010.

[27] Leroy Hood, James R Heath, Michael E Phelps, and Biaoyang Lin. Systems biology and new technologies enable predictive and preventative medicine. *Science*, 306(5696):640–643, 2004.

[28] Stuart Kauffman. Homeostasis and differentiation in random genetic control networks. *Nature*, 224:177–178, 1969.

[29] William L Lanier and S Vincent Rajkumar. Empiricism and rationalism in medicine: can 2 competing philosophies coexist to improve the quality of medical care? In *Mayo Clinic Proceedings*, volume 88, pages 1042–1045. Elsevier, 2013.

[30] Ritwik Layek, Aniruddha Datta, Michael Bittner, and Edward R Dougherty. Cancer therapy design based on pathway logic. *Bioinformatics*, 27(4):548–555, 2011.

[31] Michael Lee, Albert S. Ye, Alexandra K. Gardino, Anne Heijink, Peter Sorger, Gavin Macbeath, and Michael Yaffe. Sequential application of anti-cancer drugs enhances cell death by re-wiring apoptotic signaling networks. *Cell*, 149:780–794, 05 2012.

[32] Pey-Chang Kent Lin and Sunil P Khatri. Application of Max-SAT-based ATPG to optimal cancer therapy design. *BMC Genomics*, 13(Suppl 6):S5, 2012.

[33] Joseph Loscalzo. *Network medicine*. Harvard University Press, 2017.

[34] Hugues Mandon. *Algorithmes pour la prédiction de stratégies de reprogrammation cellulaire dans les réseaux Booléens.* Theses, Université Paris-Saclay, November 2019.

[35] Hugues Mandon, Stefan Haar, and Loïc Paulevé. Temporal reprogramming of Boolean networks. In Jérôme Feret and Heinz Koeppl, editors, *Computational*

Methods in Systems Biology - 15th International Conference, CMSB 2017, Darmstadt, Germany, September 27-29, 2017, Proceedings, volume 10545 of *Lecture Notes in Computer Science*, pages 179–195. Springer, 2017.

[36] Hugues Mandon, Cui Su, Stefan Haar, Jun Pang, and Loïc Paulevé. Sequential reprogramming of Boolean networks made practical. In *International Conference on Computational Methods in Systems Biology*, pages 3–19. Springer, 2019.

[37] Hugues Mandon, Cui Su, Jun Pang, Soumya Paul, Stefan Haar, and Loïc Paulevé. Algorithms for the Sequential Reprogramming of Boolean Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 16, Issue 5:1610–1619, 2019.

[38] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[39] David Murrugarra, Alan Veliz-Cuba, Boris Aguilar, and Reinhard Laubenbacher. Identification of control targets in Boolean molecular network models via computational algebra. *BMC Systems Biology*, 10(1):94, 2016.

[40] Jérémie Pardo, Sergiu Ivanov, and Franck Delaplace. Sequential reprogramming of biological network fate. In *International Conference on Computational Methods in Systems Biology*, pages 20–41. Springer, 2019.

[41] Jérémie Pardo, Sergiu Ivanov, and Franck Delaplace. Sequential reprogramming of biological network fate. *Theoretical Computer Science*, 872:97–116, 2021.

[42] Soumya Paul, Cui Su, Jun Pang, and Andrzej Mizera. A decomposition-based approach towards the control of boolean networks. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 11–20, 2018.

[43] Soumya Paul, Cui Su, Jun Pang, and Andrzej Mizera. An efficient approach towards the source-target control of boolean networks. *IEEE/ACM transactions on computational biology and bioinformatics*, 17(6):1932–1945, 2019.

[44] Loïc Paulevé, Juraj Kolčák, Thomas Chatain, and Stefan Haar. Reconciling qualitative, abstract, and scalable modeling of biological networks. *Nature communications*, 11(1):1–7, 2020.

[45] Loïc Paulevé and Adrien Richard. Static analysis of boolean networks based on interaction graphs: a survey. *Electronic Notes in Theoretical Computer Science*, 284:93–104, 2012.

[46] Loïc Paulevé and Sylvain Sené. Non-deterministic updates of Boolean networks. In *Proceedings of AUTOMATA'21*, volume 90 of *OASIcs*, pages 10:1–10:16. Schloss Dagstuhl Publishing, 2021.

[47] Wolfram Research. Barabasialbertgraphdistribution, 2010. [Accessed: 06-April-2022].

[48] Adrien Richard. Positive and negative cycles in boolean networks. *Journal of theoretical biology*, 463:67–76, 2019.

[49] F. Robert. Iterations sur des ensembles finis et automates cellulaires contractants. *Linear Algebra and its Applications*, 29:393–412, 1980. Special Volume Dedicated to Alson S. Householder.

[50] F Robert. Discrete iterations, a metric study. translated by rokne j, 1986.

[51] François Robert. *A Metric Tool*, pages 27–41. Springer Berlin Heidelberg, Berlin, Heidelberg, 1986.

[52] François Robert. *Les systemes dynamiques discrets*, volume 19. Springer Science & Business Media, 1995.

[53] Nidhi Sahni, Song Yi, Quan Zhong, Noor Jailkhani, Benoit Charloteaux, Michael E. Cusick, and Marc Vidal. Edgotype: a fundamental link between genotype and phenotype. *Current opinion in genetics & development*, 23(6):649–657, 2013.

[54] Mau-Hsiang Shih and Jian-Lang Dong. A combinatorial analogue of the jacobian problem in automata networks. *Advances in Applied Mathematics*, 34(1):30–46, 2005.

[55] Kyle Strimbu and Jorge a Tavel. What are Biomarkers? *Current Opinion in HIV and AIDS*, 5(6):463–466, 2011.

[56] David C Swinney and Jason Anthony. How were new medicines discovered? *Nature reviews Drug discovery*, 10(7):507–519, 2011.

[57] René Thomas. Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, 42(3):563–585, 1973.

[58] Harry L Trentelman, Anton A Stoorvogel, and Malo Hautus. *Control theory for linear systems*. Springer Science & Business Media, 2012.

[59] Gretchen Vogel. Reprogramming cells. *Science*, 322(5909):1766–1767, 2008.

[60] William M Webb. Rationalism, empiricism, and evidence-based medicine: a call for a new galenic synthesis. *Medicines*, 5(2):40, 2018.

[61] Jorge G.T. Zañudo and Réka Albert. Cell fate reprogramming by control of intracellular network dynamics. *PLoS Computational Biology*, 11(4):e1004193, 2015.

[62] Quan Zhong, Nicolas Simonis, Qian-Ru Li, Benoit Charloteaux, Fabien Heuze, Niels Klitgord, Stanley Tam, Haiyuan Yu, Kavitha Venkatesan, Danny Mou, Venus Swearingen, Muhammed a Yildirim, Han Yan, Amélie Dricot, David Szeto, Chenwei Lin, Tong Hao, Changyu Fan, Stuart Milstein, Denis Dupuy, Robert Brasseur, David E Hill, Michael E. Cusick, and Marc Vidal. Edgetic perturbation models of human inherited disorders. *Molecular Systems Biology*, 5(321):321, 2009.

Notation Index

$\bar{C}_X$	Set of uncontrollable variables of a Boolean control network.
C_X	Set of variables controllable to 1, 0 or uncontrolled of a Boolean control network.
μ, μ_i, ν	Control input representing an interpretation of the control parameters U.
μ^T	Control input where all control parameters are controlled to 1 or 0.
$\mu[k]$	Sequence of k control inputs.
s, s^i	A state of the studied Boolean network dynamics.
$[s]$	Instantiation of the equivalence class *i.e.*, $\bar{C}_X$-variables in s.
$s_{\mathcal{A}}$	Instantiation of the variables of the set $\mathcal{A}$ in s.
S_α	Set of initial states of a CoFaSe problem.
S_ω	Set of target states of a CoFaSe problem.
$s \xrightarrow{\mu} s'$	Transition from s to s' in the dynamics of F_μ.
$s \xrightarrow{\mu}{}^{*} s'$	Path from s to s' in the dynamics of F_μ.
$STBL_{F_\mu}(s)$	Predicate returning true if s is a stable state in the dynamics of F_μ.
$\mathbb{c}, \mathbb{c}_i$	Set of variables of a cycle.
$\mathfrak{c}$	Set of variables of a virtual cycle.
$\{\gamma_{\mathbb{c}}^{\top}, \gamma_{\mathbb{c}}^{\bot}\}$	A pair of opposite instantiations of a cycle $\mathbb{c}$.
$\gamma_{\mathbb{c}}^{\top}, \gamma_{\mathbb{c}}^{\bot}$	Instantiation of a cycle $\mathbb{c}$ set to one of its opposite instantiations.
$\Gamma_{\mathbb{c}}^{\top\bot}$	Set of all opposite instantiations of a cycle $\mathbb{c}$.
$\Gamma_{\mathbb{c}}^{\odot}$	Set of all instantiations of a cycle $\mathbb{c}$ that do not belong in $\Gamma_{\mathbb{c}}^{\top\bot}$.
$\gamma_{\mathbb{c}}^{\odot}$	Instantiation of a cycle $\mathbb{c}$ that do not belong in $\Gamma_{\mathbb{c}}^{\top\bot}$.
$\mathcal{A}^{\swarrow}$	Set of variables upstream of the set of variables $\mathcal{A}$ in the Boolean network interaction graph.
$\mathcal{A}^{\searrow}$	Set of variables downstream of the set of variables $\mathcal{A}$ in the Boolean network interaction graph.
$\mathcal{A}^{\times}$	Set of variables disconnected from the set of variables $\mathcal{A}$ in the Boolean network interaction graph.
$\mathcal{V}$	Set of variables belonging belonging to the SCCs of the $\bar{C}_X$-interaction graph.
$\mathcal{V}^{\swarrow_{\bar{C}_X}}$	Set of $\bar{C}_X$-variables upstream of the set of variables $\mathcal{V}$ in the Boolean network interaction graph.
$\mathcal{V}^{\diamond}$	Union of $\mathcal{V}^{\swarrow_{\bar{C}_X}}$ and $\mathcal{V}$.
$\mathcal{V}^{\Downarrow}$	Set of strictly downstream and disconnected variables of $\mathcal{V}$ in the $\bar{C}_X$-interaction graph.