Cellular automata for the observation of complex systems

Théo Plenet

THÈSE

Pour obtenir le grade de

Docteur

Délivrée par

Université Perpignan Via Domitia

Préparée au sein de l'école doctorale Énergie Environnement
Et de l'unité de recherche Espace Dev

Spécialité : **Mathématiques Appliquées**

Présentée par : **M. Théo PLÉNET**

Cellular Automata for the Observation of Complex Systems

Soutenue le 7 décembre 2022 devant le jury composé de

M. Enrico FORMENTI	Professeur, I3S, Université Nice-Sophia Antipolis	Rapporteur
M. Gilles ROUSSEL	Professeur, LISIC, Université du Littoral Côte d'Opale	Rapporteur
M. Franco BAGNOLI	Full Professor, Dép. de Physique, Université de Florence	Examinateur
M. Bastien CHOPARD	Professeur, Dép. d'Informatique, Université de Genève	Examinateur
M. David DEFOUR	Professeur, LAMPS, Université de Pepignan	Examinateur
Mme Amira MOUAKHER	Maîtresse de Conférence, Espace-Dev, Université de Perpignan	Examinatrice
M. Clément RAÏEVSKY	Maître de Conférence, LCIS, Univ. Grenoble Alpes	Examinateur
Mme Samira EL YACOUBI	Professeure, Espace-Dev, Université de Perpignan	Directrice de thèse
M. Laurent LEFÈVRE	Professeur, LCIS, Univ. Grenoble Alpes	Co-Directeur de thèse

Acknowledgements

"Never gonna give you up, Never gonna let you down"[1] *are words that I have not heard, but that I have understood through the actions of all the people who have accompanied me during these three years. I would like to thank all the people who have contributed – directly or indirectly – to the accomplishment of my thesis work, materialised by the present document.*

I would first like to thank my thesis supervisors, Samira El Yacoubi, Laurent Lefèvre and Clément Raïevsky, who have accompanied and advised me during these three years of thesis. Their complementarity on the scientific level offered me a space of great freedom while ensuring a rigorous work. Their availability, patience and involvement allowed me to work efficiently while answering my questions, calming my fears and helping me to discover the different aspects of the profession of academic researcher. I frankly couldn't have hoped for a better supervision.

I would like to thank Enrico Formenti and Gilles Roussel for agreeing to review this thesis. Their careful readings of this thesis and their different profiles provided complementary reports that brought new perspectives on this work. I would also like to thank Bastien Chopard and David Defour for accepting to be a member of the supervision committee which ensured the good progress of the thesis every year. I finally thank Amira Mouakher and Franco Bagnoli who joined the previous ones to complete my thesis jury.

I would like to thank the members of the IMAGES research team, at Espace Dev, who welcomed me during this thesis. I would like to thank in particular my colleagues, for their discussions – scientific, administrative and others – over coffee: Pius Nwachukwu, Maude Loireau, Marie Simon-El Jai, Abdelhaq El Jai and Yves Maurissen.

I would like to thank all the PhD students I have met and exchanged with during these three years. Whether it was through the Perpignan PhD students' association, through academic mobility or through online exchanges. I would like to thank in

[1] Lyrics from the refrain of Rick Astley's song *Never Gonna Give You Up*.

particular Florian (both of them), Pius, Christelle, Hanane, Caroline and Arthur.

I would like to thank Franco Bagnoli and Duccio Fanelli for welcoming me to the University of Florence, as well as all the people I met there, both for their rich scientific contributions and for their great hospitality.

I would like to thank all the people who supported me despite not fully understanding my work. A special thought for Flo and her community.

Finally, I would like to thank my family and friends for their presence and support throughout the thesis, without whom it would never have been possible.

To all those I have mentioned here and to those I have forgotten . . . Thank you.

Copyright permissions

Copyright of the template

Cellular Automata for the Observation of Complex Systems

In this thesis, we propose to study the observability of cellular automata (CA), *i.e.* how we can efficiently reconstruct the state of a CA from a limited number of measurements. To do so, we draw our inspiration from the notion of observability in classical control theory. This notion guarantees that the state of a system can be perfectly reconstructed from the measurements.

In order to apply results in classical control theory to CA, we adopt the CA definition of El Yacoubi (2008) and use it to define the notions of measurements and sensor networks for CA. Utilising these definitions we then develop the concepts of observability, reconstructibility, and adaptability in the context of cellular automata.

Subsequently, we define analytical criteria to verify these three notions for specific families of CA. We start with additive and affine CA for which we transpose the rank condition developed by Kalman and adapt it to provide similar conditions for reconstructibility and adaptability. Next, we focus on non-linear CA. We optimise observability and reconstructibility criteria for Boolean networks. We also propose a method for decentralising the observability analysis which may be applied to check observability and reconstructibility for very large CA.

We propose then another observation method with a totally different approach based on synchronisation. It avoids the computational complexity issues encountered using previous methods. In particular, in the case of small initial error, we propose an improved synchronisation method that drastically increases observation performance.

We conclude the thesis by illustrating the developed methods on three examples. We apply reconstructibility and synchronisation methods to a forest fire propagation model. We compare the observation performance of both methods. Then, we use reconstructibility and decentralised reconstructibility on a road traffic model. Finally, we use the observability of additive CA to reconstruct a sequence of numbers produced by a CA-based random number generator.

Keywords: Cellular Automata – Observation – Observability – Mobile sensors

Automates cellulaires pour l'observation de systèmes complexes

Dans cette thèse, nous nous proposons d'étudier l'observation des automates cellulaires (AC), c'est-à-dire comment nous pouvons efficacement reconstruire l'état d'un AC à partir d'un nombre limité de mesures. Pour cela, nous puisons notre inspiration dans la théorie du contrôle classique, en particulier dans la notion d'observabilité. Celle-ci garantit que l'état du système peut être reconstruit parfaitement à partir des mesures.

Afin d'appliquer les résultats de la théorie classique du contrôle aux AC, nous adoptons la définition des AC présentée par El Yacoubi (2008) et l'utilisons pour définir les notions de mesures et de réseaux de capteurs. En utilisant ces définitions, nous développons ensuite les concepts d'observabilité, de reconstructibilité et d'adaptabilité dans le contexte des automates cellulaires.

Ensuite, nous définissons des critères analytiques pour vérifier ces trois notions pour des familles spécifiques d'AC. Dans un premier temps, pour les AC additifs et affines pour lesquels transposons la condition de rang développé par Kalman. Nous adaptons cette condition pour fournir des critères similaires pour la reconstructibilité et l'adaptabilité. Ensuite, nous nous intéressons spécifiquement aux AC non linéaires. Nous optimisons un critère d'observabilité et de reconstructibilité utilisé pour les réseaux booléens et nous proposons aussi une méthode de décentralisation de l'observabilité, celle-ci permettant de vérifier l'observabilité pour des AC de très grande taille.

Nous présentons une autre méthode d'observation qui ne se base pas sur la théorie du contrôle. Celle-ci permet d'éviter les problèmes de complexité calculatoire causés par les méthodes précédentes. Nous nous intéressons en particulier au cas où l'erreur initiale est faible, ce qui permet, dans certains cas, d'augmenter drastiquement les performances d'observation.

Nous terminons en appliquant les outils développés dans cette thèse sur des exemples concrets. Nous appliquons la reconstructibilité et la synchronisation sur un modèle de propagation d'incendie de forêt. Nous en profitons pour comparer les performances d'observations de ces deux méthodes. Ensuite, nous utilisons la reconstructibilité ainsi que la reconstructibilité décentralisée sur un modèle de trafics routiers. Puis, nous utilisons l'observabilité des AC additifs pour reconstruire une séquence de nombres produite par un générateur de nombres aléatoires basé sur un AC.

Mots-clés : Automates cellulaires – Observation – Observabilité – Capteurs mobiles

Publications

Publications in peer reviewed journals

[1] **Plénet, T.**, El Yacoubi, S., Raïevsky, C., and Lefèvre, L. (2022). Observability and reconstructibility of bounded cellular automata. *International Journal of Systems Science*, 1-17.

[2] **Plénet, T.**, El Yacoubi, S., Raïevsky, C., and Lefèvre, L. (2022). Observability and Reconstructibility of Affine Cellular Automata: Example on Random Number Reconstruction. *Journal of Cellular Automata*.

[3] El Yacoubi, S., **Plénet, T.**, Dridi, S., Bagnoli, F., Lefèvre, L., and Raïevsky, C. (2021). Some control and observation issues in cellular automata. *Complex Systems*, 30(3), 391-413.

Conference Papers (with presentations)

Name of the author with † indicates the presenter.

[1] **Plénet, T.**†, El Yacoubi, S., Raïevsky, C., Lefèvre, L., and Bagnoli, F. (2022). Synchronisation of Elementary Cellular Automata with a Small Initial Error. Application to Rule 18. *In International Conference on Cellular Automata for Research and Industry* (pp. 73-82). Springer, Cham.

[2] **Plénet, T.**†, El Yacoubi, S., Raïevsky, C., and Lefèvre, L. (2020, December). Observability of affine cellular automaton through mobile sensors. *In International conference on cellular automata for research and industry* (pp. 36-45). Springer, Cham.

[3] **Plénet, T.**†, Raïevsky, C., Lefèvre, L., and El Yacoubi, S. (2020). Social Organisation of Mobile Sensors for Wildfire Spread Estimation. *IFAC-PapersOnLine*, 53(2), 3596-3601.

Long French Abstract

La modélisation des phénomènes naturels est un enjeu crucial tant pour la compréhension du monde qui nous entoure que pour la prédiction d'événements futurs. D'autant plus que le monde subit des changements majeurs dus au changement climatique. Il devient urgent de développer de telles méthodes pour faire face à ces problèmes. Ces approches peuvent être appliquées dans un large éventail de domaines et d'applications différentes. Ceci encourage le développement d'une formalisation théorique afin de faciliter le déploiement de ce type d'outil dans le futur. La thèse va dans ce sens et fournit également quelques exemples appliqués.

Pour donner un peu de contexte, intéressons-nous aux feux de forêt dans le sud de la France. Entre 2015 et 2019, plus de 8 700 feux de forêt ont été signalés dans la zone méditerranéenne française[2]. Selon une récente étude interinstitutionnelle sur l'impact du changement climatique sur les risques d'incendie de forêt en France (Chatry *et al.*, 2010), le risque d'incendie de forêt, actuellement concentré sur la zone méditerranéenne de la France, s'étendrait à la grande majorité de la France et particulièrement au nord de la France d'ici 2070. Il est donc nécessaire de fournir aux autorités des méthodes efficaces pour estimer le risque d'incendie et pour suivre l'évolution des incendies lorsqu'ils se produisent.

Pour développer de telles méthodes, il est nécessaire d'avoir une représentation du phénomène étudié. Cette représentation est appelée « modèle » et est construite à partir de données quantitatives exprimées en termes formels. De cette représentation, on déduit des observations possibles qui sont ensuite confrontées à des résultats expérimentaux. Elle peut être exprimée à l'aide d'équations mathématiques, ou au moyen d'autres descriptions plus adaptées.

Pour les systèmes physiques répartis dans l'espace, les modèles utilisent largement les équations aux dérivées partielles (EDP) comme outil de représentation. Les EDP sont un ensemble d'équations mettant en relation les dérivées de variables physiques par rapport à l'espace et au temps. Elles ont été largement utilisées par les physiciens depuis plusieurs siècles. Cependant, le développement de tels modèles d'EDP prend souvent beaucoup de temps et nécessite des connaissances expertes dans le domaine d'application considéré, mais aussi un solide bagage en mathématiques et en informatique. L'existence de solutions et les simulations numériques sont déjà des

[2] Données de Prométhée, qui étudie les feux de forêts en Corse, en région PACA, en région Languedoc Roussillon ainsi qu'en Ardèche et dans la Drôme. Plus d'informations sur https://www.promethee.com

problèmes très complexes en soi. Tous ces problèmes impliquent généralement des simplifications radicales du modèle. Malgré cela, seules des solutions numériques sont réalisables. Elles nécessitent généralement des efforts de calcul intensifs et une expertise en analyse numérique.

Lorsque l'utilisation des EDP devient trop compliquée, d'autres méthodes implémentent directement l'espace et le temps de manière discrète peuvent être utilisées. Dans ce contexte, les automates cellulaires (AC) apparaissent comme une bonne alternative, car l'espace et le temps sont représentés de manière discrète, mais aussi, car seule la dynamique locale du phénomène est représentée. La dynamique globale est obtenue comme la superposition de toutes les dynamiques locales.

Organisation du manuscrit

L'objectif de cette thèse est de présenter les automates cellulaires (AC) comme un outil d'observation des systèmes complexes. Le manuscrit est organisé en 6 chapitres qui présentent le contexte, l'ensemble des travaux, ainsi que quelques exemples d'application. Le premier chapitre sert d'introduction en présentant le contexte et l'originalité des travaux. Le deuxième sert de définition aux AC et aux différentes notions liées à l'observation. Le troisième chapitre présente des critères analytiques permettant de vérifier les trois notions définies précédemment. Le quatrième chapitre présente une méthode d'observation alternative basée sur la synchronisation des AC. Le cinquième chapitre illustre les travaux précédents au travers de trois exemples simples. Le dernier chapitre conclut cette thèse en donnant un aperçu des perspectives et en exposant les travaux futurs qui pourraient découler des recherches effectuées dans cette thèse.

Chapitre 2

Le chapitre 2 présente les bases nécessaires à la compréhension des autres chapitres. Nous commençons par présenter la définition formelle aux automates cellulaires (El Yacoubi, 2008). À partir de celle-ci, nous présentons les différentes règles de transition que nous utiliserons dans les chapitres suivants. Parmi ces différentes règles, les plus utilisés dans le reste du manuscrit sont les automates cellulaires élémentaires de Wolfram et les automates cellulaires additifs et affines. Dans la seconde partie du chapitre, nous décrivons différentes notions liées à l'observation des automates cellulaires. Les notions d'observabilité et de reconstructibilité sont des notions déjà existantes que nous avons adaptées aux automates cellulaires. L'adaptabilité, au contraire, est une nouvelle notion facilitant le lien entre l'observabilité et la reconstructibilité.

Chapitre 3

Dans le chapitre 3, nous donnons des critères qui permettent de vérifier l'observabilité, la reconstructibilité et l'adaptabilité pour certaines familles d'automates cellulaires. Dans un premier temps, nous étendons le critère d'observabilité classique de Kalman (Kalman, 1960) aux automates cellulaires additifs et affines. Nous donnons aussi un critère de reconstructibilité et d'adaptabilité pour cette même famille. Ensuite, nous utilisons la représentation par graphe de transition d'état, ainsi que des résultats de l'observabilité et de la reconstructibilité des réseaux booléens (Wang *et al.*, 2012), pour proposer un algorithme permettant de vérifier l'observabilité et la reconstructibilité des automates cellulaires non linéaires (non additifs). Finalement, nous présentons une méthode d'observation des automates cellulaires qui permet de définir un critère d'observabilité régionale et de décentralisation de l'observabilité pour les automates cellulaires non linéaires.

Chapitre 4

Dans le chapitre 4, nous présentons la synchronisation des automates cellulaires et nous l'étudions comme un estimateur d'état. Cette approche est sensiblement différente de celles présentées au troisième chapitre, car la bonne reconstruction de l'état par l'estimateur n'est pas caractérisée par l'observabilité ou la reconstructibilité, mais par une étude statistique. Dans un premier temps, nous présentons la synchronisation des automates cellulaires de Bagnoli (Bagnoli *et al.*, 2010) et nous détaillons en quoi celle-ci est très proche de la formulation d'un estimateur d'état de la théorie du contrôle classique. Ensuite, nous caractérisons la propagation de l'erreur initiale dans les automates cellulaires élémentaires lorsque l'erreur initiale est faible. Finalement, à partir de cette caractérisation, nous pouvons présenter une nouvelle méthode de synchronisation lorsque l'erreur initiale est faible et nous comparons ses performances d'observation à la précédente.

Chapitre 5

Dans le chapitre 5, nous appliquons les méthodes d'observation présentées dans les chapitres précédents sur trois exemples concrets. Pour le premier exemple, nous étudions un modèle de propagation de feu de forêt. Cet exemple permet de détailler la méthode de construction d'un estimateur d'état en utilisant à la fois l'observabilité et la synchronisation d'un automate cellulaire non linéaire. Dans le second exemple, nous étudions un modèle de trafic routier et plus particulièrement le problème de

la reconstructibilité décentralisée permettant de construire un observateur d'état modulaire pour la surveillance des réseaux routiers. Nous terminons avec un exemple d'automate cellulaire affine générateur de nombres aléatoires. Nous utilisons l'observabilité pour trouver les nombres aléatoires générés par celui-ci.

Contributions Majeures

Contributions majeures La thèse propose 4 contributions majeures : la définition de l'observabilité pour les automates cellulaires, la généralisation de la condition de Kalman aux automates cellulaires additifs et affines, l'adaptation du critère d'observabilité des réseaux booléens aux automates cellulaires, et la formalisation de la synchronisation des automates cellulaires comme estimateur d'état.

Dans le chapitre 2, nous avons formalisé les notions d'observabilité, de reconstructibilité et d'adaptabilité pour les automates cellulaires. L'observabilité et la reconstructibilité ont déjà été définies pour d'autres types de systèmes par le passé (système linéaire invariant en temps, réseau booléen, etc.). La contribution de cette thèse réside dans la transposition de ces caractérisations aux automates cellulaires. Nous avons également décrit l'injectivité comparée, une généralisation de l'injectivité par rapport à une autre fonction qui est intimement liée à la notion de reconstructibilité. Ensuite, nous avons formalisé la notion d'adaptabilité issue des travaux de Laschov *et al.* (2013) pour dériver une condition nécessaire d'observabilité qui peut être vérifiée avant la construction de la séquence de sortie. Cette notion est particulièrement importante, car elle joue un rôle direct dans le placement des capteurs lorsque nous souhaitons assurer l'observabilité.

Dans le chapitre 3, nous avons généralisé la condition du rang de Kalman comme critère d'observabilité pour les automates cellulaires additives et affines. Nous avons proposé des critères similaires pour l'adaptabilité et la reconstructibilité. Il ne s'agit pas d'une condition de rang, mais sur le noyau de la matrice de sortie ou d'observation. Le critère d'adaptabilité exige que les noyaux soient disjoints et la reconstructibilité exige l'inclusion des noyaux. Les critères d'observabilité et de reconstructibilité sont également complétés par un corollaire qui prévoit la possibilité de reconstruire l'état initial ou actuel du système à partir de la séquence de sortie.

Toujours dans le chapitre 3, nous avons amélioré les critères d'observabilité et de reconstructibilité des réseaux booléens en bénéficiant des avantages des automates cellulaires. Ceci permet de développer un algorithme qui vérifie à la fois l'observabilité et la reconstructibilité pour chacune des configurations initiales. Contrairement aux autres méthodes, cet algorithme peut prendre en compte les conditions sur les configurations initiales et ainsi réduire considérablement le nombre de configurations à étudier. Ceci réduit la complexité algorithmique en offrant la possibilité d'évaluer

l'observabilité ou la reconstructibilité de l'AC avec un grand nombre de cellules. Dans le cas où cette réduction ne s'applique pas ou ne réduit pas suffisamment la complexité, nous avons présenté une méthode de décentralisation de l'analyse d'observabilité qui permet de linéariser la complexité en divisant le problème en problèmes plus petits. Cette méthode ne fonctionne pratiquement que sur l'AC unidimensionnelle, car elle nécessite un grand nombre de capteurs pour les dimensions supérieures.

Dans le chapitre 4, nous avons formalisé la synchronisation de l'AC comme estimateur d'état. Cette approche est radicalement opposée à la précédente puisque nous ne cherchons pas à vérifier au préalable si l'observation fonctionnera, mais seulement à étudier statistiquement ses performances. Nous avons également développé une méthode optimisée qui assure la coordination des capteurs mobiles si l'erreur initiale est composée d'une seule cellule erronée. Dans l'exemple de la propagation d'un feu de forêt, la synchronisation optimisée et l'approche de reconstructibilité atteignent des performances moyennes d'observation similaires.

Cette thèse pose les bases de l'observation de l'AC en fournissant des définitions de l'observation et deux méthodologies d'estimation d'état. Cependant, nous sommes encore loin du déploiement d'un réseau de capteurs utilisant des automates cellulaires pour la surveillance de systèmes physiques. Plusieurs perspectives concernant les futurs travaux qui découlent de cette thèse sont présentées dans le chapitre 6.

Contents

Nomenclature

Acronyms

ACA	Additive or Affine Cellular Automata
BN	Boolean Network
CA	Cellular Automata
DPS	Distributed Parameter Systems
ECA	Elementary Cellular Automata
PCA	Probabilistic Cellular Automata
PDE	Partial Differential Equation
RNG	Random Number Generator
STP	Semi Tensor Product

Notations

$\wedge$	logical conjunction
$\mathcal{L}$	lattice of the CA
$[\![a;b]\!]$	set of integers in interval $[a;b]$
$\neg$	logical negation
$\vee$	logical disjunction
$\ltimes$	semi tensor product
$\min(a;b)$	minimum between a and b
$\mathbb{N}$	set of natural integers
$\otimes$	Kronecker product
$\mathcal{S}$	state space of the CA
$\mathbb{Z}$	set of relative integers
A'	transpose of matrix A
$A^{\dagger}$	pseudo-inverse of matrix A
A^{-1}	inverse of matrix A

Chapter 1

Introduction, Motivations and State of the Art

The modelling of natural phenomena is a crucial issue in understanding the world around us, for the prediction of future events or for the monitoring of current events. Especially as the world is undergoing major changes due to climate change. It is becoming urgent to develop such methods to cope with climate change issues. These approaches can be applied in a wide range of different domains and applications. This encourages the development of a theoretical formalisation in order to facilitate the deployment of this kind of tool in the future. The thesis goes in this direction and provides a few applied examples as well.

To give some context, let us look at the forest fires in the south of France. Between 2015 and 2019, over 8 700 forest fires were reported in the French Mediterranean area[1], the most affected region of France. According to a recent inter-institutional study on the impact of climate change on wildfire hazards in France (Chatry *et al.*, 2010), the risk of wildfires, currently concentrated in the Mediterranean area of France, would spread to the vast majority of France and particularly to the north of France by 2070. It is therefore necessary to provide the authorities with efficient methods to estimate the risk of fires and to monitor the progress of fires when they occur.

Forecasting and monitoring methods are not only useful for monitoring forest fires but can also be used to study other systems, *e.g.* to study the spread of epidemics (Keeling and Eames, 2005), to predict desertification in sub-Saharan Africa (Koné *et al.*, 2020), or to detect fake news in social media (Zhou *et al.*, 2020). The examples are numerous and that is why we wish to study these methods theoretically without being constrained to a single type of system or field of application.

To build such methods, two elements are needed: a good knowledge of the dynamics of the phenomena and information about it at a given time. From this we can predict, with an accuracy that depends on the two elements provided, how it will evolve. Information about the system is determined through measurements, which can be satellite imagery or data from sensors placed in the environment. The knowl-

[1] Data from Prométhée, which studies forest fires in Corsica, the PACA region, the Languedoc Roussillon region as well as in Ardèche and Drôme. More information on https://www.promethee.com

edge of the dynamics, called a model, can take different forms (cellular automata, partial differential equations, etc.). These notions of *model* and *measurement*, at the core of this thesis, will be developed with more detail in the next paragraphs.

This thesis proposes different methods to build useful tools for the monitoring of natural phenomena, and more generally complex systems. Hereafter, we start by an introduction to the different concepts discussed in this thesis. To begin with, we will focus on the notion of model and what it means. We will present different forms that a model can take and we will explain why we have chosen cellular automata for our study. Then, we will present the notions of measurements and observation of a system. The notion of observation, when studied through system theory, has a particular definition that we will present in detail. Then, finally, we will present the plan of this thesis which will put forward the methods of observation that we developed.

1.1 Modelling of Complex Systems

1.1.1 System and Model

Before discussing the notions of modelling and system model, we need to introduce what a system is and what it is about.

What is a system?

A *system* is a set of components that interact with each other to form a meaningful group. Systems can represent physical, digital or even conceptual entities. As an example of a system, we could mention several systems with very different components: A car is a system where the components are the parts of the car ; The internet is a system where the components are servers, computers and communication channels ; A forest fire is a system where the components are the trees that compose the forest, the wind that helps the fire to spread, and even the firefighters that prevent the spread of the fire. There are as many systems as there are ways of looking at reality. Systems can be classified into large families that encompass similar behaviour. We will discuss only about two: complex systems and distributed parameter systems.

Complex systems represent systems with a very large number of components. They are often characterised by the emergence of phenomena that occur at the system level and cannot be explained at the component level solely. Among these systems, we find economic systems such as the stock market, ecological systems such as ant colonies or digital systems such as the internet.

Distributed parameter systems (DPS) represent systems whose components vary continuously according to different coordinates (often space and time). They are usually represented by partial differential equations (see hereafter). As an example of DPS, we could propose the propagation of a forest fire, as the forest and the wind depends on both time and position.

As stated in the title of this thesis, we will pay particular attention to complex systems and in particular to those who are also DPS. Once the system and its components have been identified, the system can be modelled. This is the first stage in studying or predicting the evolution of a system.

What is a model?

A *system model*, or more simply model, is a representation for a system. It can take different forms, either deterministic or probabilistic, continuous or discrete, linear or non-linear. In physics, the model is constructed from quantitative data expressed in formal terms. From the model one derives possible observations that are confronted with experimental results. The model itself may be expressed using mathematical equations, or by means of other descriptions that are more suited, for instance, for a computer implementation. From this, two types of models can be distinguished, *knowledge-based* models and *data-based* models. In a knowledge-based model, the model uses internal variables that represent a physical quantity, for example the temperature for a forest fire propagation model. On the opposite, in a data-based model, the variables are "hidden", the model describes the behaviour of the system without taking into account the internal states of the system. It is usually constructed from a large amount of data. Such models are very common in deep learning. And there are a large number of models that fall somewhere in between. For example, the FARSITE simulator describes the propagation of fire as elliptical, without taking into account the temperature of the fire. The physical variables are "hidden" but the global dynamics of the system are not.

When is a model considered valid?

A model is associated with a domain of validity which describes in which context and to what extent the model is valid. To show the interest of the domain of validity of a model, let us take as an example two different models of gravitation. The first is the universal law of gravitation described by Newton between 1665 and 1685 which explains that any two bodies exert a similar but opposite force which depends on the distance. The second is the law of general relativity presented by Einstein around 1915 which describes that gravity is no longer a force, but a curvature of space-time which depends on the mass. These two models describe the gravitational force but at different scales. To calculate the trajectory of a tennis ball both model

will provide a solution but Einstein's model is too sophisticated for this situation. Similarly, Newton's model would not be as accurate for modelling gravity when the gravitational field is too intense, for example near a black hole, or to explain the apsidal precession of Mercury.

What is important to remember is that a model is defined in terms of a domain of validity and that there is no point in taking a model that is too complex to solve much less complex problems. As Einstein said[2], "*Everything should be made as simple as possible, but no simpler.*".

1.1.2 Different Methods to Build a Model

As we have seen, there are many different models, but there are also many different methods of representing these models. We are going to present several knowledge-based methods to build complex or DPS models. The difference between several major types of models lies in the way the spatial phenomena in the system are represented. Indeed, they can be either represented through a continuum or discrete variable. On the one hand, the continuous solution provides a way to represent the evolution continuously with respect to time and space. On the other hand, discrete variable model are easier to handle. In the next few sections, we present 4 different methods: partial differential equations, network-based models, cellular automata, and agent-based models. The first one is a continuous one whereas the others are discrete.

In the discrete approach, the dynamics are often described locally. The system is modelled as a set of subsystems that interact with each other and not as a single large entity as in the continuous approach. This type of method is simpler to implement because scientists often describe local interactions between subsystems more easily than a whole system. The downside is that the equation that describes the system globally is not straightforward to find based on the local interactions. This problem will be investigated in the thesis since the observation of systems (see section 1.2) requires the expression of the system global dynamics.

Partial Differential Equations

Partial Differential Equations (PDE) are a set of equations relating derivatives of physical variables with respect to space and time. They have been widely used by physicists for several centuries. Examples are the Navier-Stokes equation for modelling Newtonian fluids, the Maxwell's equations for modelling electromagnetic fields or the equations governing a reaction–diffusion system. PDE can also be used to model more specific

[2] This quote is commonly attributed to Einstein, but it could be the paraphrase of an idea attributed to Einstein. More on this at https://quoteinvestigator.com/2011/05/13/einstein-simple/

systems: Rothermel (1972) uses PDE to model the propagation of fires ; Black and Scholes (2019) use PDE to describe the dynamics of the stock market.

However, the development of such PDE models often takes a long time and requires expert knowledge in the considered application field, but also a strong background in mathematics and numerics. Existence of solutions and numerical simulations are already very complex problems by their own, not to mention analysis of their dynamical properties. All those problems typically involve radical model simplifications. Even so, only numerical solutions are achievable. They typically require intensive computational efforts and numerical analysis expertise. To overcome this, we can use methods that directly implement space and time in a discrete manner.

Network-Based Models

Network-based models are characterised by the presence of a number of components (nodes) that are only connected to a certain part of the other components. The behaviour of these components is then defined locally from the interactions made with the other components of the network. This type of model is widely used to model systems that are already present as networks such as social networks or computer networks. It can also be used to model other types of systems such as epidemics where two people are connected on the network if there is a possibility that they will transmit the disease (*e.g.* through in-person interaction).

Boolean networks are a particular type of network-based model. The components are logical variables (usually Boolean). Boolean networks are particularly used in bioinformatics where they are used to model the interactions between genes in the DNA. We will discuss the properties of Boolean networks in section 3.3, as an introduction to the study of observability for non-linear cellular automata.

Agent-Based Models

Agent-based models represent models where system components are modelled by agents. Each agent is an entity that interacts with other agents, but unlike network-based models, there is no constraint on how agents can interact. It is also possible to add a social aspect to the agents so that they function as an organisation, this is called a multi-agent system. This type of model is rather difficult to study at the system level because the interactions between the agents may not follow predetermined formal rules.

Cellular Automata

Cellular automata (CA) are a special case of network-based models. The components, called cells, are placed on a lattice, often uniform, and interact with nearby compo-

nents only. This makes it easy to model systems with local spatial dynamics. For example, CA can be used to model the dynamics of fluids through the interaction of particles in the fluid (Hardy *et al.*, 1973). In addition, CA are also widely used in the modelling of a wide variety of systems, from the modelling of clotting processes (Ouared and Chopard, 2005) to the modelling of desertification in sub-Saharan Africa (Koné *et al.*, 2020). Chopard and Droz (1998) provides numerous tools on the modelling of physical systems through cellular automata.

Several CA, with different domains of validity, can be used to model the propagation of a forest fire. The simplest one we can think of is to represent the evolution of the fire by discrete states: *tree*, *burning tree* and *burnt tree*. The fire spreads from one cell to its neighbours. Karafyllidis and Thanailakis (1997) propose a more complex model that uses the burnt biomass as a state, which can take into account different parameters such as wind or topography. CA are also used in more complex fire propagation simulators such as FlamMap (Finney, 2006). These simulators are used by firefighters to anticipate fire spread and act accordingly. This simulator simulates several variables such as fire, wind or humidity that interact together.

Another advantage of cellular automata is the ease of parallelisation. Indeed, the computation of the state of each cell depends only on the state of the neighbouring cells at the previous time. Therefore, it is very easy to parallelise the computation of each cell, as for agent-based models. When the neighbourhoods used by the CA are regular then it is possible to parallelise on GPU which increases even more the possibility of parallelisation and allows a gain of performance when it comes to the speed of calculation compared to the PDE.

Translated with www.DeepL.com/Translator (free version)

1.2 Observation of Complex Systems

Now that we understand the notion of system, we can make the observation of it. But first what is observation? In this context, we define the observation as the action of gradually reconstructing the state of the system on the basis of measurements. Measurements represent partial information on the system taken by sensors. There are two main types of reconstruction, one based on data and the other on model. The first one uses a large number of measurements and process the data to extract the state of the system. Such methods are used for instance with Wireless Sensor Network or Big Data. The other method uses a smaller number of measurements but takes advantage of a model of the system to carry out the reconstruction. Such methods are used in control theory to build controllers which are widely used in different fields of engineering.

In this thesis, we will focus on the second method, more specifically on control

theory approaches and how they can be applied to CA. In the next few paragraphs, we will present a short history of control theory, the main problems of studying DPS through control theory and the state of current research on control theory for CA.

1.2.1 Control Theory and Distributed Parameters System

Initially, control theory referred to the regulation of a physical process. In this sense, it has existed for several centuries (*e.g.* the centrifugal regulators of James Watt created in 1788). This thesis refers instead to the recent formalisation of control theory that emerged in the 1950s based on dynamical systems representation (state space or external). Among the works of this time, the best known are those of Kalman (1960) on linear systems with notably the definition of controllability and observability. Roughly speaking, controllability guarantees that there exists a regulator for this system and observability guarantees that the state of the system can be reconstructed correctly from measurements. It is the latter that we will study in this thesis. Since then, control theory has become widely known in a field of engineering known as control engineering.

Control theory can study linear or non-linear systems, continuous or discrete time, with one or more inputs. The study of DPS, however, remains largely open. The spatial dimension poses various problems related to infinite dimensional space and operators. Roughly speaking, this means studying a spatial continuum of time-dependent variables. In addition to the computational difficulties of studying PDE, many results of classical control theory do not apply to DPS directly. For example, the notion of observability is separated into two different notions: an exact observability and an approximate observability. Exact observability is similar to usual observability, whereas approximate observability guarantees that the state of the system can be reconstructed only approximately Glowinski and Lions (1995). The two notions coincide in the case of classical systems.

In some cases, the observability of the system can only be assured for a part of the system. El Jai *et al.* (1993) present a regional study of observability and controllability that provides methods to study only a small region of the system. Controllability and observability therefore have particular definitions that relate to the region under study. Even if the verification is more complicated, it still ensures partial observability where it would not otherwise be possible.

1.2.2 Control Theory of Cellular Automata

There is little work addressing the control theory of CA. El Yacoubi (2008) proposes a definition of cellular automata that encompasses the notions of actuators and control.

The regional controllability is presented and illustrated in the context of an additive CA. Regional controllability has also been studied by Dridi *et al.* in different ways. In (Dridi *et al.*, 2020), it is evaluated using graph theory and Hamiltonian paths, in (Dridi *et al.*, 2019) it is checked using a Markov chain. The studies of these two authors focus on regional controllability where, contrary to the cases of classical theory, the duality between controllability and observability is not trivial. In (Urıas *et al.*, 1998), the observability of an additive CA is studied in a hidden way. The authors place sensors so as to ensure that a matrix is invertible, which is a hidden application of the Kalman condition.

Alternatively, if we constrain the definition of CA to bounded and finite state CA then it can be studied as a Boolean network. The control theory of Boolean networks has been widely studied since the early 2000s with contributions from Cheng (2005) on the semi-tensor product and Laschov *et al.* (2013) on the observability graph.

Nonetheless, CA exhibit certain advantages compared to PDE. Indeed, they represent the spatial dimension in a discrete way. Consequently, for a finite space, the number of variables is finite. This leads to the use of the results of the classical finite dimensional control theory in the case of a DPS. In particular, it is possible to use Kalman rank condition on controllability and observability.

1.3 Contribution and Content of the Thesis

In this thesis, we will focus on the observation of complex systems and in particular DPS. PDE are very powerful tools for the study of such systems, but their studies can be very difficult. Among discrete systems, CA seem to be the most suitable models for modelling DPS, due to the strong spatial component. CA are models where the time and space are discrete. We will use them as a model for DPS. Therefore, the systems modelled by CA are not strictly DPS but will have the same behaviour regarding their validity domain. Although CA are less precise than PDE, their domain of validity will be sufficient for the purpose of the thesis, which is the observation of spatially distributed physical systems. This thesis contributes to the observation of complex systems in three ways:

- firstly with a formal definition of observability, reconstructibility and adaptability, three notions related to the reconstruction of the state of a system from measurements. Observability and reconstructibility are two notions that we have applied to cellular automata while adaptability is a new notion presented in this thesis that provides a way to verify that the sensors are well suited for the observation of the system.

- secondly, by providing analytical criteria to verify the previous notions. For

additive and affine CA, the Kalman condition for observability is adapted for CA and similar criteria are given for reconstructibility and adaptability. For non-linear CA, the criteria are derived from the observation of Boolean networks. They are adapted to take advantage of the benefits of CA in order to reduce the algorithmic complexity. A decentralised observability and reconstructibility verification method is given to further reduce the complexity in the case of CA with a large number of cells.

- thirdly, by studying the synchronisation of CA as a state estimator. This method is not derived from control theory and does not depend on observability or reconstructibility. Its efficiency is statistically evaluated through a large number of simulations. An efficient synchronisation method is given for the observation of a system where the initial error is small.

The manuscript is organised in 6 chapters that present the contributions, examples and conclusions on future work.

Chapter 2 serves as a definition for CA and notions related to observation. It presents in detail the definition of CA by El Yacoubi as well as different properties of CA that will be used in the sequel. The notions of observability, reconstructibility and adaptability are presented and serve as a basis for the control theory of CA.

Chapter 3 presents analytical criteria to verify the three notions defined previously. First, criteria are given for additive and affine CA by generalising the Kalman rank condition. Then, criteria are given for non-linear CA which are studied as Boolean networks. Finally, in the last part of this chapter, methods are presented to handle the algorithmic complexity raised by the criteria through the decentralisation of the verification for observability.

Chapter 4 presents an alternative observation method based on CA synchronisation. We present synchronisation as a state estimator for CA observation. We then present a more efficient method for CA observation when the initial error is very small.

Chapter 5 presents three examples which aim to illustrate the methods presented in this thesis on simple examples. The first example deals with a forest fire propagation model that we use to compare the performance of reconstructibility against synchronisation. Secondly, we present a road traffic model for which we try to apply the proposed observation decentralisation approach. We end with an example on the reconstruction of random numbers generated by a CA random number generator, which allows us to describe the use of the methods on additive or affine CA.

Chapter 6 concludes this thesis by providing an overview of the prospects and outlining future work that might follow on from the research carried out in this thesis.

Chapter 2

Cellular Automata as a Modelling Tool

Résumé

L'objectif de ce chapitre est de présenter les bases nécessaires à la compréhension des autres chapitres. Dans ce but, nous présenterons les ***automates cellulaires*** *comme un outil de modélisation des systèmes complexes, mais aussi les notions d'****observabilité*** *et de* ***reconstructibilité****, liées à l'observation de ceux-ci. Dans une première partie, nous définirons les automates cellulaires ainsi que les différentes règles de transition que nous utiliserons dans la suite du manuscrit. Dans la seconde partie, nous présenterons les notions de capteurs, d'observabilité, de reconstructibilité et d'adaptabilité, liées à l'observation des automates cellulaires.*

Les principales contributions de ce chapitre sont les définitions proposées d'observabilité, de reconstructibilité et d'adaptabilité, adaptées aux automates cellulaires. Ces définitions s'appliquent au cas de l'observation par un réseau de capteurs, fixes ou mobiles.

Contents

2.1 Introduction

The observation of a complex system may be realised through the use of a mathematical description of the process to be observed. This description is a partial representation of the reality written with mathematical equations and is called a *mathematical model* or a *system model*. To describe a system (whether complex or not) with such a mathematical model, it is necessary to have both a working knowledge of its parts and a precise idea of the applications for which the model will be used. The model is thus intimately related both to the system and its application.

For complex systems (*i.e.* with a large number of subsystems), the use of PDE becomes too complicated. The different time and spatial scales, as well as the different variables of the subsystems, make it extremely difficult to establish equations that describe the whole system. In this context, cellular automata (CA) appear as a good alternative because only the local dynamics (*i.e.* relating the variables characterising neighbour subsystems) are used as equations. The global dynamics is obtained as the superposition of all the local dynamics.

The objective of this chapter is to present CA as a tool for modelling complex systems and to give all the theoretical tools necessary to understand this work. We start with a short history of CA followed by a mathematical formalisation of these. Then, we present the elements that make a CA a good tool for modelling complex systems and finally, we describe the tools necessary to observe complex systems modelled by CA.

2.2 Generalities on Cellular Automata

2.2.1 Brief History of Cellular Automata

The history of CA begins in the 1940s with the collaboration of John Von Neumann and Stanislaw Ulam, two mathematicians from the National Laboratory of Los Alamos in New Mexico (USA). John Von Neumann was working on self-replicating systems and Stanislaw Ulam on crystal growth using a grid. This collaboration led to the creation of the first cellular automaton: the "*universal copier and constructor*", a self-replicating system that uses a two-dimensional grid.

CA were not extensively studied in the 40s and 50s due to the lack of computing power provided by computers at this period. It is only from the end of the 60s that CA start to be widespread in the mathematical community, notably thanks to the works of Gustav A. Hedlund, who studied CA as a dynamic system in its own right.

It is in the 70s and 80s that the foundations of today's cellular automata are laid with notably the works of: John Horton Conway on the game of life (Conway

et al., 1970); Stephen Wolfram on elementary cellular automata (Wolfram, 1983); and Hardy, Pomeau, and De Pazzis on the HPP model (Hardy *et al.*, 1973). John Horton Conway proposed the game of life, a CA defined by extremely simple rules but able to show complex global behaviour (notably proven to be a universal Turing Machine). Hardy, Pomeau, and De Pazzis developed the HPP model, a CA capable of simply modelling the behaviour of fluids through the interaction of its particles, which would later become lattice gas automata (Wolf-Gladrow, 2004, Chap 3) or lattice Boltzmann methods (Wolf-Gladrow, 2004, Chap 5). Stephen Wolfram published in the 80s a series of systematic studies on elementary CA (the simplest ones) showing the existence of different types of behaviour and allowing classifying elementary cellular automata according to their properties.

Finally, it is from the beginning of the 90s that cellular automata start to be widespread as a tool for modelling physical (Chopard and Droz, 1998) or biological systems.

2.2.2 *Definition of Cellular Automata*

There are many definitions of CA, both formal and informal, which vary from one application to another and from one domain to another. Among all these definitions, there are two central points. The first is the "cellular" aspect of the CA, whereby the space is divided into elementary parts called cells and each of these has a state that describes its nature. The second point concerns the evolution of the state of the cells over time. The evolution of a cell's state depends both on the state of this cell and on the states of neighbouring cells. This rough definition allows us to consider the use of CA for applications involving a large number of entities that interact locally (complex systems), in fluid mechanics (local interaction of the particles making up the flow of the fluid) or in the representation of spatially distributed transport phenomena (such as forest fire spread), just to cite a few examples.

Our objective is to formally define cellular automata as a tool for the observation of complex systems. We will therefore use a definition of CA adapted to the observation problem, namely the one proposed by El Yacoubi (2008). The author proposes therein a definition of bounded cellular automata suitable for control. In addition to this definition, the corresponding state equation, as well as the notions of actuators and regional controllability, are also presented. We will detail these notions in section 2.3.1, when considering the duality between observation and control problems. For the time being, we restrict ourselves to the definition 2.1 of bounded cellular automata.

Definition 2.1 (Cellular Automata)**.** A **bounded cellular automaton** is defined by a quadruple $\mathcal{A} = (\mathcal{L}, \mathcal{S}, \mathcal{N}, f)$ where:

- $\mathcal{L}$ is a d-dimensional finite lattice of cells (a generic cell will be denoted c, in the sequel) which are spatially arranged according to their shape. We denote by $N \in \mathbb{N}$ the number of cells in $\mathcal{L}$.

- $\mathcal{S}$ denotes a finite set of admissible state values for the cells. We usually represent a set of k-states by the finite commutative ring $\mathcal{S} = \{0, 1, \ldots, k-1\}$ which uses modular arithmetic. $\mathcal{S}$ may also be written as $\mathcal{S} = \mathbb{Z}/k\mathbb{Z}$.

- $\mathcal{N}$ is a mapping which defines the cell's neighbourhood. This neighbourhood is usually the same for all cells but it can vary through space and time. It is given by:

$$\begin{aligned} \mathcal{N} \colon \mathcal{L} &\to \mathcal{L}^n \\ c &\mapsto \mathcal{N}(c) = \{c_1, c_2, \ldots, c_n\} \end{aligned} \tag{2.1}$$

  where the cells c_i, for $i \in [\![1, n]\!]$, are linked to the cell c by an influence relation. In most of the case, the cell c is in its neighbourhood. In the case of bounded cellular automata, it is necessary to take into account the lattice boundary conditions (see Figure 2.2) in the construction of the neighbourhoods.

- f is a transition function which determines the cell's state at time $t+1$ given the state of the neighbouring cells at time t. It is defined by:

$$\begin{aligned} f \colon \mathcal{S}^n &\to \mathcal{S} \\ s_t(\mathcal{N}(c)) &\mapsto s_{t+1}(c) = f(s_t(\mathcal{N}(c))) \end{aligned} \tag{2.2}$$

  where $s_t(c)$ represents the state of the cell c at time t and $s_t(\mathcal{N}(c)) = \{s_t(c') | c' \in \mathcal{N}(c)\}$ is the state of the cells in the neighbourhood at time t.

This definition, although very general, largely limits the possible applications with this kind of CA. The bounded lattice removes the universal Turing machine property and the finite number of states hinders the representation of continuous state CA such as the forest fire spread model of Karafyllidis and Thanailakis or the Lattice Boltzmann methods, both discussed in the introduction.

In the following subsections, we clarify the different components presented in definition 2.1. We present the most common lattices, neighbourhoods and boundary conditions that are used in CA.

Global Transition Function F

In some cases, in particular to solve the observation problem (see section 2.3.3), we need to model the local transition function f as a so-called *global* transition function. It applies simultaneously to all the cells of the CA and not only to the neighbouring cells. It is used for example to define the state transition diagram or the state representation, two very important notions that we will use in chapter 3.

Therefore we start by defining the state of all cells in the CA, $s_t \in \mathcal{S}^{\mathcal{L}}$, called the **configuration** at time t. It is defined by the application :

$$\begin{aligned} s_t : \mathcal{L} &\to \mathcal{S} \\ c &\mapsto s_t(c) \end{aligned} \tag{2.3}$$

The transition from one configuration to another is determined by the global transition function F defined by :

$$\begin{aligned} F : \mathcal{S}^{\mathcal{L}} &\to \mathcal{S}^{\mathcal{L}} \\ s_t &\mapsto F(s_t) = s_{t+1} \end{aligned} \tag{2.4}$$

The transition function f describes the local evolution (*i.e.* at the cell level) of the CA, whereas the global transition function F describes the global evolution (*i.e.* at the configuration level). However, the transformation from one transition function into the other is not a simple problem. One of the first issues in solving the observation problem will be to obtain the global transition function F because most CA are defined through the local transition function f.

Due to the finiteness of the state space $\mathcal{S}$ and the lattice $\mathcal{L}$ of the CA, we provide the definition 2.2 which allows us to represent the configuration s_t by a number in base k. For example, $230012_{|4}$ represents the configuration s_t such that $s_t(c_1) = 2$, $s_t(c_2) = 3$ and so on. This configuration corresponds to the number 2822 in base 10.

Definition 2.2 (Numerical Notation of Configuration)**.** By arranging the cells of the lattice in any given arbitrary order, we can represent the configuration s_t by an N-digit number in base k, denoted by $s_{t|k}$. It also has a unique equivalent in base 10.

Lattice of Cells $\mathcal{L}$

In the vast majority of cases, the topology of the lattice is regular, *i.e.* it consists of a single shape and size of cell to cover the space. In the one-dimensional case, the cells are points arranged in a line, but it is usual to represent them as a line of square cells. In the two-dimensional case, several regular lattices exist, the most commonly used

being the square, triangular or hexagonal cell lattices. The use of one lattice or another can change the possible dynamics of the modelled system. For example, in the case of the HPP Model (Wolf-Gladrow, 2004, Chap. 3/sec. 1), the lattice of square cells allows the particles to spread in 4 directions, whereas in the FHP Model (Wolf-Gladrow, 2004, Chap 3/sec. 2), 6 directions can be available by using a lattice of triangular cells.

Set Space $\mathcal{S}$

In definition 2.1, the state of the cell $\mathcal{S}$ is presented as a finite set of natural integer $\mathcal{S} = \{0, 1, \ldots, k-1\}$. In chapter 3, when we discuss the observability of additive and affine CA, we will need to perform some mathematical operations on these states (in particular the inverse of the modular multiplication). Therefore, it will be necessary to consider $\mathcal{S}$ as a finite field (or Galois field), in this case a finite commutative ring with a prime number of states k.

However, in some specific cases discussed in the introduction, the state of the cells is not an integer but a word (*tree* or *burning tree* as in the trivial example of the fire spread model), a colour (*black* or *white* as in the game of life) or any other discrete value. In those particular cases where the state of the cells is not represented by integers, values of the state space $\mathcal{S}$ are simply arbitrary values which correspond to the states of the cells. For example, in Conway's Game of Life (Conway *et al.*, 1970), colours *white* and *black* can be respectively represented by the integer 0 and 1. definition 2.1 can still represent CA that do not have integer states (as described in the definition).

Neighbourhood $\mathcal{N}$

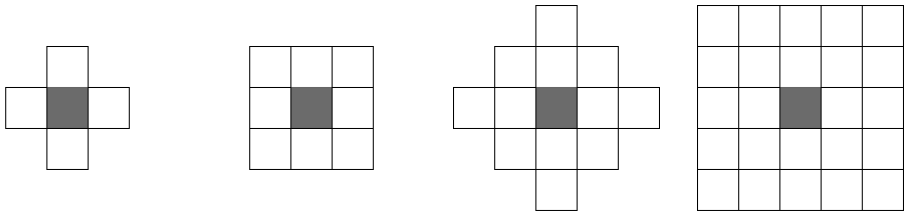
Most of the work on CA use uniform neighbourhood, that is the neighbourhoods are identical for all the cells in the lattice and defined through a distance to the central cell. If some cells in some particular neighbourhood have no influence on the future state of the considered cell, then the transition function will be chosen appropriately, while the neighbourhood definition itself remains the same for all cells. Therefore, we can define the neighbourhood formally by the following relation:

$$\begin{aligned} \mathcal{N} &: \mathcal{L} \to \mathcal{L}^n \\ & c \mapsto \mathcal{N}(c) = \{c' \in \mathcal{L} \,|\, \|c' - c\|_i \leq r\} \end{aligned} \tag{2.5}$$

where $\|c\|_i$ represents the i-norm. Among the possible norms, the most used are the taxicab norm ($i = 1$) or the infinity norm ($i = \infty$).

These two norms allow different cells to be included in the neighbourhoods, norm 1 includes only cells that are adjacent by a side whereas norm ∞ includes cells adjacent by a side but also by a corner. This difference is visible, on Figure 2.1, in the case of a two-dimensional mesh with square cells.

In the one-dimensional case, these two norms are equivalent because there are no corner-adjacent cells. In the case of a two-dimensional lattice with square cells, these two norms correspond to the Von Neumann ($i = 1$) and Moore ($i = \infty$) neighbourhoods. Figure 2.1 shows different neighbourhoods with different distances for 2D lattices with square cells. For a hexagonal cell lattice, these two neighbourhoods are equivalent (as for a 1D square cell lattice) and we use the term uniform neighbourhood.



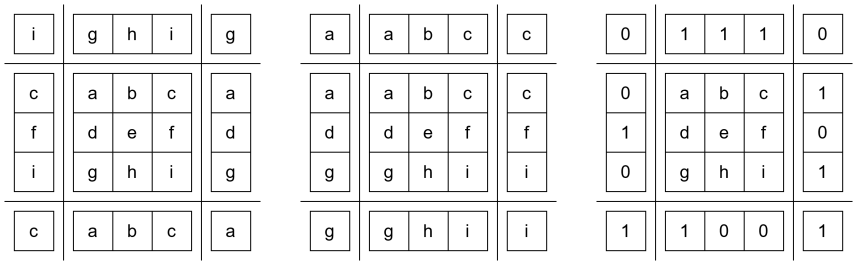
(a) $i = 1; r = 1$ (b) $i = \infty; r = 1$ (c) $i = 1; r = 2$ (d) $i = \infty; r = 2$

FIGURE 2.1. Different neighbourhoods for two-dimensional lattice of square cells as a function of the i-norm and the distance r. Neighbourhoods from left to right: Von Neumann of distance 1; Moore of distance 1; Von Neumann of distance 2; and Moore of distance 2.

Boundary Conditions of the Lattice

In the case of a bounded CA (such as in definition 2.1) the lattice is finite and the neighbourhood of the cells on the border depends on cells that do not exist in the lattice. We must therefore consider the value of the cells directly outside the lattice. These are called boundary cells. In addition, since some physical systems admit natural boundaries, it is natural to be able to define these boundaries in their CA models. This is known as a boundary condition. Three types of boundary conditions are usually considered: **periodic**, **reflexive**, and **fixed** (see Figure 2.2).

- **Periodic** boundary conditions allow the system to be looped back on itself, whereby the state of the boundaries on one side corresponds to the state of the cells on the other side of the lattice. In the two-dimensional case, a lattice with periodic boundaries is in fact the surface of a torus.
- **Reflexive** boundary conditions correspond to the situation where the cells located on the boundary of the lattice have themselves as neighbours.
- **Fixed** boundary conditions correspond to boundary cells that have a value that is determined by the CA designer. This value can vary from one cell to another and can also vary as a function of time (known as a boundary trajectory). In some applications it is also referred to as a **null** boundary condition, which corresponds to a fixed boundary condition of 0. In many cases it has a special meaning, for instance, the "void" or "quiescent" state.



(a) Periodic Boundary Conditions (b) Reflexive Boundary Conditions (c) Fixed Boundary Conditions

Figure 2.2. Examples of the boundary conditions for a two-dimensional cellular automaton with a 3×3 lattice of square cells. The letters from a to i represent the cells of the CA and the cells on either side of the CA represent the boundary cells.

In some particular cases, especially when we address the decentralisation of observability (see section 3.4 on page 71), we are only interested in one region of the CA. To study the observability of this region, we model it by a separate CA, while the rest of the outer cells are only considered through the boundary conditions. They do not respect any of the three types of conditions defined previously as they are simply the state of cells of the larger CA. These are called here **unknown** boundary conditions.

2.2.3 *Families of Cellular Automata*

To carry out a systematic study, which applies to all CA, proves to be an impossible task. Therefore, the properties of CA are often studied for some specific families which are obtained by introducing constraints on the definition. One of the earliest and most influential studies of this kind was conducted on elementary cellular automata (CA with one-dimensional lattices, Boolean state space, and neighbourhood of radius 1) by Wolfram (1983). More recently, several studies by Chaudhuri *et al.* have been published in a book (Chaudhuri *et al.*, 1997) gathering results on additive cellular automata (CA with linear transition function). Many other studies have been conducted on other families of CA such as totalistic CA by Wolfram (1984) or two-dimensional CA by Packard and Wolfram (1985).

In this section we will introduce many families, but we shall concentrate on the one we will consider later. We will define these families through the definition 2.1, which will allow us to use them in the following chapters of this thesis.

Wolfram's Elementary Rules

The elementary rules of Wolfram have been defined in Wolfram (1983) with the purpose of performing a systematic study of the behaviour of CA. These rules are

considered elementary because they are the simplest non-trivial CA: the lattice is one-dimensional, the state space has 2 state (Boolean state space) and the neighbourhood of distance 1 (it includes only the two directly adjacent cells and the cell itself). Based on the definition 2.1, Wolfram's elementary cellular automata (ECA) are defined by:

- $\mathcal{L} = [\![0; N-1]\!]$ is a one-dimensional lattice of N cells.
- $\mathcal{S} = \{0, 1\}$ is a Boolean state space. In a Boolean state space, we have the possibility to use Boolean or modular arithmetic. In this case, we will use three operators (logical AND "$\cdot$", logical OR "+" and logical NOT "$\overline{a}$") for the Boolean arithmetic and two operators (product modulo 2 "$\cdot$" and addition modulo 2 "$\oplus$") for the modular arithmetic. The logical AND and the multiplication modulo 2 are equivalent as well as the addition modulo 2 "$\oplus$" and the logical XOR.
- $\mathcal{N}: c_i \mapsto \{c_{i-1}, c_i, c_{i+1}\}$ is the neighbourhood of distance 1 which includes the central cell and its two direct adjacent cells. By default, we use the periodic boundary conditions but any other boundary conditions can be used.

No specific constraint has been placed on the transition function f as the purpose of Wolfram was to study the behaviour of every possible transition function. In a similar way to a ternary logical operator, the transition function f is defined by a truth table that maps the state, at time t, of the 3 neighbouring cells to the state, at time $t+1$, of the central cell (see Table 2.1). Since each of the three cells in the neighbourhood have a Boolean state, there are $2^3 = 8$ entries in the truth table and they are associated with a digit that represents the state of the central cell at the next time step. The transition function of the ECA is therefore described by these 8 digits. Each of the possible transition functions can be represented by an 8-bit number called *Wolfram's elementary rule* number. These numbers are arranged in descending order of the integer number whose development in base 2 is obtained from the binary values of the 3 neighbouring cells (*i.e.* the neighbourhood value 101 corresponds to the integer value 5), as shown in Table 2.1. An example of the evolution of this rule is given in Figure 2.3 for 15 cells and periodic boundaries.

$s_t(\mathcal{N}(c_i))$	111	110	101	100	011	010	001	000
$s_{t+1}(c_i)$	0	1	0	1	1	0	1	0

TABLE 2.1. Truth table of the transition function of the Wolfram's ECA 90, or simply rule 90. The first row represents the possible states of the neighbourhood at time t and the second row represents the state of the central cell at time $t+1$ corresponding to the neighbourhood of the column.

The notation "rule 90", which will be abbreviated to **R90**, gives all the information about the behaviour of the transition function but nothing about the size of the lattice or the boundary conditions. Therefore, we define a more detailed notation **RX-N-B**

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

FIGURE 2.3. Example of configuration evolution of the ECA of Table 2.1. The cells on the ends are the boundary condition cells (here periodic condition) used in the neighbourhood calculation.

which includes not only the rule number **X**, but also the number of cells **N** as well as the nature of the boundary conditions **B** as described in Table 2.2. In addition, it is possible to refer to **RX-N**, **RX-B** or **RX** if the number of cells or the nature of the boundaries are not specified.

Boundary condition	Null	Fixed	Periodic	Reflexive	Unknown
Notation B	NB	FB	PB	RB	UB

TABLE 2.2. The boundary conditions notation for writing ECA in the form **RX-N-B** where **B** is the term dedicated to boundary conditions. For example, the ECA **90** of Figure 2.3 with **15** cells and periodic boundaries has the notation **R90-15-PB**.

Although initially defined by a truth table, the transition functions of ECA can also be written in algebraic form using either modular arithmetic (see eq. (2.6)) or Boolean arithmetic (see eq. (2.7)). These algebraic forms, although equivalent, are used in different scenarios depending on their ease of use. For example, if one is interested in Boolean derivatives of transition functions then modular arithmetic is preferred and if one is attempting to emulate CA with logic gates then Boolean arithmetic is preferred.

Example 2.1

The following are the two algebraic forms of **R90** defined by the truth table in Table 2.1. The first uses modular arithmetic and the second Boolean arithmetic.

$$s_{t+1}(c_i) \mapsto s_t(c_{i-1}) \oplus s_t(c_{i+1}) \tag{2.6}$$

$$s_{t+1}(c_i) \mapsto (s_t(c_{i-1}) \cdot \overline{s_t(c_{i+1})}) + (\overline{s_t(c_{i-1})} \cdot s_t(c_{i+1})) \tag{2.7}$$

Although these CA are elementary, they are not without interest for the modelling of complex systems. In mathematics for instance, ECA can be used for the generation of fractal figures like the Sierpiński triangle with **R126** on Figure 2.4 (other rules like **R60**, **R90**, **R150** generate variants of this figure). **R110** (with an infinite lattice) has the property of being Turing Complete (Cook *et al.*, 2004), *i.e.* **R110** can simulate Turing machines where the writing of the algorithm is carried out in the choice of the initial configuration. Physical systems can also be modelled using these ECA: **R30** can generate the figures naturally present on the shells of the *Conus Textile*[1] (see

[1] More information on the *Conus textile* can be found on https://en.wikipedia.org/wiki/Conus_textile

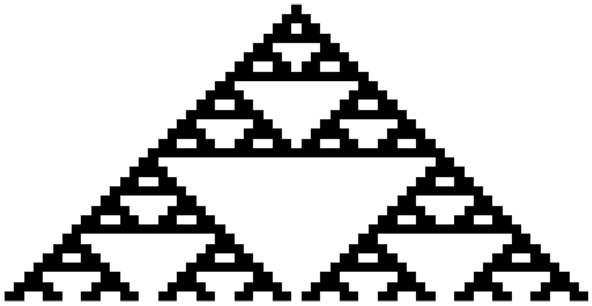


FIGURE 2.4. Generation of the Sierpiński triangle with **R126**. Evolution of **R126-50-PB** from a single black cell. The black cells represent 1 and the white cells represent 0. Time is going down on the vertical axis.

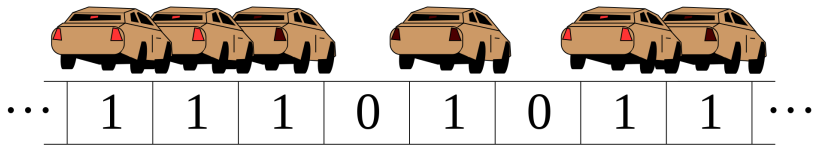
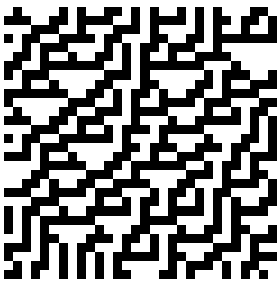


FIGURE 2.5. **R184** interpreted as a simulation of traffic flow. Each 1 cell corresponds to a vehicle, and each vehicle moves forward only if it has open space in front of it. Image from David Epstein, English Wikipedia.

Figure 2.6), a species of sea snail; **R184** can model a very simple road network (see Figure 2.5) or the deposition of particles on a surface (Chopard and Droz, 1998).



(a) **R30-30-PB**

(b) *Conus Textile*

FIGURE 2.6. Example of the evolution of **R30-30-PB** for a time horizon of 30 and a picture of the *Conus Textile* shell. Image from Richard Ling on Wikipedia.

Additive Cellular Automata

Additive cellular automata (ACA) have first been used to study the algebraic properties of CA (Martin *et al.*, 1984). In this paper, Martin *et al.* use ACA to study the state transition diagram and the length of the cycle in the graph, we discuss that point on section 2.2.4. They have since been used in different ways, such as a random number generator (Tomassini *et al.*, 2000), or a hash function (Chaudhuri *et al.*, 1997, Chap. 8).

Many of the possible applications of ACA have been compiled in the book *"Additive cellular automata: theory and applications"* by Chaudhuri, Chowdhury, Nandi, and Chattopadhyay. The element that bounds together all these CA is the nature of their transition function. Indeed, the transition function is defined as a weighted sum of the states of the neighbouring cells. Using definition 2.1, the transition function f can then be defined as:

$$\begin{aligned} f &: \mathcal{S}^n \to \mathcal{S} \\ s_t(\mathcal{N}(c_i)) &\mapsto s_{t+1}(c_i) = \sum_{j=1}^{n} a_j \cdot s_t(c_i^{(j)}) \end{aligned} \tag{2.8}$$

where $c_i^{(j)}$ is the j^{th} cell in the neighbourhood of the cell c_i as shown in Figure 2.7, a_j is the weight applied to the state of the cell $c_i^{(j)}$, and n is the number of cells in the neighbourhood.

Note that the i does not represent the cell number but the position of the cell in the neighbourhood. Indeed, the weights of the transition function are defined in relation to the position of the cell within the neighbourhood, as shown in Figure 2.7, and not in relation to a particular cell in the CA.

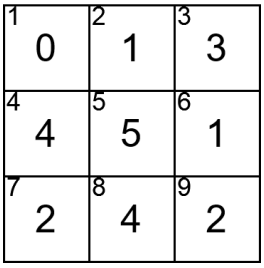


FIGURE 2.7. Example of the weights applied for a 2D additive cellular automaton with a Moore neighbourhood and a state space with (at least) 6 states. The number in the upper left corner of each cell represents the j number of that cell in the neighbourhood while the central number represents the a_j weight applied in the sum.

Remark 2.1. Two-dimensional ACA (or one-dimensional ACA represented in the form of a 2D ACA of size $1 \times N$) can be computed through a 2D convolution using the neighbourhood with the weights (as in Figure 2.7) as the convolution filter. The CA must still be augmented by representing the boundary conditions as cells, so that they are taken into account in the convolution.

Since the transition function is linear over the state of all cells in the neighbourhood, f is therefore a linear form[2] over the commutative ring $\mathcal{S}$. Using the properties

[2] A short definition of linear forms can be found on Wikipedia. https://en.wikipedia.org/wiki/Linear_form

of linear form, the local transition equation f can be written with a matrix product between the row vector a of the sum weights and a column vector composed of the states of the neighbourhood cells.

$$s_{t+1}(c_i) = \begin{bmatrix} a_1 & a_2 & \dots & a_n \end{bmatrix} \cdot \begin{bmatrix} s_t(c_i^{(1)}) \\ s_t(c_i^{(2)}) \\ \vdots \\ s_t(c_i^{(n)}) \end{bmatrix} \tag{2.9}$$

By slightly adapting this function, we are able to describe the evolution of the state of a cell as a function of the configuration, *i.e.* the state of all the cells of the CA and not only those of the neighbourhood. To do this, we apply the weight 0 on the cells that are not in the neighbourhood. Moreover, we note $x_t \in \mathcal{S}^N$ the vector representation of the configuration $s_t \in \mathcal{S}^{\mathcal{L}}$. For consistency with definition 2.2 on the numerical notation of configurations, the same order of cells must be taken for the arrangement of x_t and $s_{t|k}$. Therefore, we obtain the equation (2.10) which describes the evolution of the cell c_i as a function of the configuration x_t.

$$s_{t+1}(c_i) = A_i \cdot x_t = \begin{bmatrix} a_{i,1} & a_{i,1} & \dots & a_{i,1} \end{bmatrix} \cdot \begin{bmatrix} s_t(c_1) \\ s_t(c_2) \\ \vdots \\ s_t(c_N) \end{bmatrix} \tag{2.10}$$

where $a_{i,j}$ is the weight applied to the state of cell c_j in the calculation of the state of c_i. If the cell c_j is in the neighbourhood ($c_j \in \mathcal{N}(c_i)$) then the value of $a_{i,j}$ is the weight associated to its position in the neighbourhood and it is 0 otherwise.

Having formulated the evolution of the state of a cell according to the configuration s_t, we can build the global transition function F which uses the matrix A which is composed of the rows $A_i, \forall i \in [\![1; N]\!]$.

$$\begin{aligned} &F \colon \mathcal{S}^N \to \mathcal{S}^N \\ &x_t \mapsto x_{t+1} = A \cdot x_t = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,N} \\ a_{2,1} & a_{2,2} & \dots & a_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N,1} & a_{N,2} & \dots & a_{N,N} \end{bmatrix} \cdot \begin{bmatrix} s_t(c_1) \\ s_t(c_2) \\ \vdots \\ s_t(c_N) \end{bmatrix} \end{aligned} \tag{2.11}$$

This matrix expression of the global transition function F will allow us to apply the results from linear system theory to CA, as discussed in chapter 3.

Affine Cellular Automata

Additive cellular automata represent only a small fraction of the possible CA. For example, only 8 of the 256 ECA are additive (**R0**, **R60**, **R90**, **R102**, **R150**, **R170**, **R204**, **R240**) knowing that **R0** and **R204** are respectively the null and identity CA. Some of Wolfram's ECA are linear with respect to the Boolean negation of a cell. This is notably the case of **R195** which is defined by $s_{t+1}(c_i) = s_t(c_{i-1}) \oplus \overline{s_t(c_i)}$. In Boolean arithmetic, the NOT operation ($\overline{a}$) is equivalent to the XOR operation (or addition modulo 2) with 1, so the negation operation is represented by the addition of the constant 1. Therefore, in order to include these CA, and more generally every ACA with constants, we can adapt the definition by adding a constant vector $\eta \in \mathcal{S}^N$ in the transition function (2.11) which then reads:

$$\begin{aligned} F\colon \mathcal{S}^N &\to \mathcal{S}^N \\ x_t &\mapsto x_{t+1} = A \cdot x_t + \eta \end{aligned} \tag{2.12}$$

We refer to these CA as **affine cellular automata** (or additive with a constant).

In the following, we will use the term ACA to designate both affine and additive cellular automata. Indeed, the results of chapter 3 apply to affine CA and thus also to the particular case of affine CA with a null constant, additive CA. We could simply use the term affine CA, but we wish to use both terms as additive CA is widely used within the CA community whereas the term affine (and linear to refer to affine with null constant) is more used in the system theory community.

Hybrid Cellular Automata

Some cellular automata, notably the Tomassini *et al.* (2000) random number generator studied in chapter 5, have a local transition function f that varies from one cell to another. Such cellular automata are known as hybrid CA. We can adapt definition 2.1 by adding a dependency on the cell position to the transition function. Then f_i will be defined as:

$$\begin{aligned} f_i\colon \mathcal{S}^n &\to \mathcal{S} \\ s_t(\mathcal{N}(c_i)) &\mapsto s_{t+1}(c_i) = f_i(s_t(\mathcal{N}(c_i))) \end{aligned} \tag{2.13}$$

In the case of hybrid ECA, the notation $\langle 90, 160, 253, 126\rangle$ is used (Choudhury *et al.*, 2009) to describe the rule of each cell of the CA. In this case, the first cell follows **R90**, the second **R160** and so on. Using the notation of Table 2.2, we can replace the number of the rule **X** and the number of cells **N** by the notation $\langle\cdot,\cdot,\cdot\rangle$. Therefore, if the hybrid ECA $\langle 90, 160, 253, 126\rangle$ has periodic boundaries, we can write **R⟨90,160,253,126⟩-PB**.

For additive and ACA, the transition function, defined by (2.8) is described by the weights a_j and the constant η. Thus, the weights will depend on both the position of the cell c_j in the neighbourhood and on the position of the cell c_i on the lattice. Therefore the transition function is defined by :

$$\begin{aligned} f_i &: \mathcal{S}^n \to \mathcal{S} \\ s_t(\mathcal{N}(c_i)) &\mapsto s_{t+1}(c_i) = \sum_{j=1}^{n} a_{i,j} \cdot s_t(c_i^{(j)}) + \eta_i \end{aligned} \tag{2.14}$$

Probabilistic Cellular Automata

In some cases it is necessary to use probabilities when modelling a physical system. For example, probabilities are used to model a fire outbreak in a forest fire spread model or to reproduce an accident in a road traffic model. We must then transform the transition function of the CA into a random process. Therefore, probabilistic CA (PCA) may be defined in many different ways. We will use the formal definition from Mairesse and Marcovici (2014).

In the probabilistic case, the local transition function f does not allow computing the deterministic value for the state of the cells at the next time, but instead the probability distribution of a random variable with values in $\mathcal{S}$. We will denote $\mathcal{M}(\mathcal{S})$, the set of variables with random value in $\mathcal{S}$. Consequently, the global transition function F does not provide a particular configuration, but instead a random variable with value in $\mathcal{M}(\mathcal{S}^{\mathcal{L}})$. It will be defined as:

$$F : \mathcal{M}(\mathcal{S}^{\mathcal{L}}) \to \mathcal{M}(\mathcal{S}^{\mathcal{L}})$$

The vast majority of the definitions and results given in the rest of this thesis do not apply to PCA. For example, for a given configuration, there are several possible subsequent configurations, which makes many results, such as observability and reconstructibility (see section 2.3.3), irrelevant. The different observability and reconstructibility criteria given in chapters 3 and 4 will therefore not apply to PCA. On the other hand, other definitions and results may be easily adapted to PCA, like those on the state transition graph (see section 2.2.4 on *State Transition Diagram and Attractor*) which can be considered, in the probabilistic case, as the graph representation of a Markov process with a well-defined corresponding Markov matrix (once the transition probabilities have been defined). Furthermore, PCA are discussed in section 3.4 where they are used as a tool for observing deterministic CA.

2.2.4 *Properties of Cellular Automata*

Some properties are common to many (if not all) CA such as reversibility, state transition diagrams and complexity classes are directly related to the observability of CA. These will be used throughout this work and are defined in this section.

State Transition Diagram and Attractor

In the case of the CA definition 2.1 , the number of cells is finite as is the number of states for a cell. In addition, since the configuration s_t is the application that associates a cell with its state, the number of configurations is also finite. Configurations are ordered arrangements of N cells with values from the set S (with k states), so a total of k^N possible configurations. Moreover, with the finiteness of the space of configurations $\mathcal{S}^{\mathcal{L}}$, it is possible to represent the global transition function F as a map which connects one configuration to another (which may be itself). By representing this lookup table as a binary relation (this representation is detailed in section 3.3 on page 54), we can define a graph (see Figure 2.8) from the adjacency matrix. This graph is called the **State Transition Diagram** and was first presented for CA by Martin *et al.* (1984). In addition, in their paper, Martin *et al.* use this diagram to illustrate the existence of attractor cycles. After a certain number of iterations and whatever the initial configuration, the state of the CA is in a cycle. The number and size of the cycle depend on every parameter of the CA.

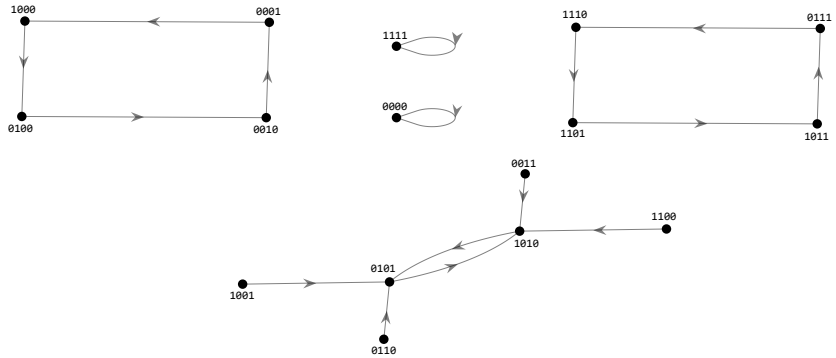


Figure 2.8. State transition diagram of the elementary Wolfram cellular automaton **R184-4-PB**. This CA has 5 different cycles: 2 of size 1, 2 of size 4, and 1 of size 2.

The state transition diagram represents the transition function as a graph and therefore make using graph theory results on CA possible. Using the numerical notation of the configurations s_t (see definition 2.2), it is possible to number each node of the state transition diagram and thus define an adjacency matrix A and a state vector x_t that represents the configurations. For example, the configuration 0000010 of ECA

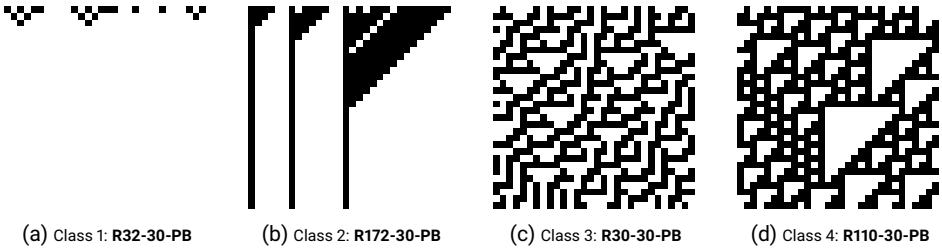
has 2 for decimal value and is represented by the vector $x_t = \begin{bmatrix}0 & 0 & 1 & 0 & \dots & 0\end{bmatrix}'$. The first element of x_t corresponds to the configuration with the value 0, the second to the configuration 1, and so on.

In this work, the notion of attractors and cycles was not explored but could have been used as a criterion of observability or reconstructibility as it was done by Laschov *et al.* (2013) (more details in 3.3.1). Hamiltonian circuits were used as a criterion for controllability of CA by Dridi *et al.* (2020) and the transmission, through duality, of such results to observability remains an open question, and not necessarily a trivial one.

Classes of Cellular Automata

In 1984, Wolfram proposed a classification of CA into four different classes. Initially studied with totalistic CA (CA whose evolution is described by the sum of the states of the cells of the neighbourhood as in the game of life), Wolfram extends these classes to every CA. They are defined according to the behaviour of the CA, for the vast majority of initial configurations, after a certain number of iterations. These classes are defined by :

- **Class 1**: Initial configurations evolve into a stable, homogeneous, and unique configuration for all cells. **R32** shown in Figure 2.9a is an example of a class 1 ECA.
- **Class 2**: Initial configurations evolve into a stable configuration or an oscillating pattern (which is neither homogeneous nor unique). This class serves as a "filter" for the randomness of the initial configuration whereby only a few structures remain, the others being filtered out. **R172** shown in Figure 2.9b is an example of a class 2 ECA.
- **Class 3**: Initial configurations evolve into a chaotic or pseudo-random (ergodic) pattern: no stable or repetitive state is reached. Wolfram defines a criterion of spatial $d^{(x)}$ and temporal $d^{(t)}$ chaoticities that quantify the chaoticity of the CA. Wolfram also proves that the injectivity of F (reversibility) is a sufficient (but not necessarily) condition for maximal chaoticity $d^{(x)} = 1$. **R30** shown in Figure 2.9c is an example of a class 3 ECA.
- **Class 4**: Initial configurations evolve into different structures, which may be similar to other classes, interacting in complex ways. Wolfram conjectured that some of the rules would be Turing Complete (have the ability to simulate any Turing machine) which was proven for the rule **R110** (Cook *et al.*, 2004) and Conway *et al.*'s Game of Life. **R110** shown in Figure 2.9d is an example of a class 4 ECA.



(a) Class 1: **R32-30-PB** (b) Class 2: **R172-30-PB** (c) Class 3: **R30-30-PB** (d) Class 4: **R110-30-PB**

Figure 2.9. Example of evolution for ECA to show the behaviour of different classes of CA presented by Wolfram. Time is going down on the vertical axis for 30 time step.

Reversibility of Cellular Automata

Reversibility is a property of cellular automata that describes the possibility of finding the previous configuration s_{t-1} from the current configuration s_t. This property must apply for all configurations of the CA. Reversibility is identical to the injectivity of the global transition function F which ensures, for each configuration, the uniqueness of its antecedent.

Some configurations have no predecessors, they exist only as initial configurations. These configurations are called *gardens of Eden* and are related to the surjection of the transition function F. Moore (1962) and Myhill (1963) define the Garden of Eden's theorem which binds the injection and surjection of CA, that is the non-existence of the Garden of Eden to the notion of reversibility.

Reversiblity is notably used for applications such as random number generation, hash tables, etc. In the case of Cellular Automata Random Number Generator (CA RNG) reversibility is used with a maximum cycle size to ensure equiprobability of the configurations (the random numbers) of the CA. An example of CA RNG is described in more detail in the chapter 5: Application to Monitoring of Complex Systems (see section 5.4 on page 129) where the notion of observability is used to retrieve information about the random number generator flow.

2.3 Observability and Reconstructibility of Cellular Automata

The objective of the observation of a CA is to determine its internal state (*i.e.* its configuration s_t) using only partial measurements made on it. For this purpose, we will use a state estimator (or state observer). This is a system that has its own state (known to the user) and which we seek to converge to the actual state of the system. The first step is to choose the approach for estimating the state, but also to ensure that it works well for all (or for a large part of) the possible scenarios.

In classical control theory, it is possible to characterise whether or not there is a state estimator to reconstruct the state of the system given the dynamics of the system

and the dynamics of the measurements. This characterisation, called **observability**, is given by Kalman and can be easily evaluated for linear systems. For non-linear systems, the verification is more challenging as it requires a deeper understanding of dynamical systems. Furthermore, when the time is discrete, Kalman observability separates into two different notions: observability and reconstructibility. The former characterises the reconstruction of the initial state of the system (*i.e.* s_0) and the latter the reconstruction of the current state (*i.e.* s_T).

In this section, we will start with a formalisation of the notion of measurement and sensors for AC. This will allow us to define observability and reconstructibility for CA. We will also present a third characterisation called adaptability that serves as a link between these two notions.

2.3.1 State Representation of Cellular Automata

In classical control theory, the state representation is used to represent the studied system by including the notion of input and output. The best-known state representation is the one describing a linear time-invariant (LTI) system:

$$\begin{cases} x_{t+1} &= Ax_t + Bu_t \\ y_t &= Cx_t + Du_t \end{cases} \tag{2.15}$$

x_t, y_t and u_t are vectors that respectively represent the state, the output and the input of the system at time t. x is called the state vector, y the output vector and u_t the input vector. A, B, C and D are matrices that described the behaviour of the system. A is called the state matrix which represents the dynamic of the system. B is the input matrix which describes how the input impact the system. C is the output matrix that describes how the system is measured. And D is the feedforward matrix which represents the direct impact of the input on the output.

Within the scope of this work, we are only interested in the observation of systems, so there is no input (*i.e.* $u = 0$), the simplified state representation is:

$$\begin{cases} x_{t+1} &= Ax_t \\ y_t &= Cx_t \end{cases} \tag{2.16}$$

In many cases, the considered system is not linear, so the linear state representation cannot work. The state representation for non-linear systems may be given as:

$$\begin{cases} x_{t+1} &= f(x_t) \\ y_t &= h(x_t) \end{cases} \tag{2.17}$$

where the map f describes the evolution of the state (state dynamics), while h is the output map.

Equation (2.18) describes the non-linear state representation adapted for the observation of CA. The state of the system is the configuration s_t and its evolution is determined by the global transition function F. The output operator H represents the sensor network and θ_t represents the measurements taken by the sensors at time t.

$$\begin{cases} s_{t+1} &= F(s_t) \\ \theta_t &= H(s_t) \end{cases} \tag{2.18}$$

With this representation, the state of the CA at time t can be calculated from the initial configuration s_0 using F^t which represents the composition of F t times:

$$\begin{cases} s_t &= F^t(s_0) \\ \theta_t &= H(F^t(s_0)) \end{cases} \tag{2.19}$$

The definition of H and θ_t is the subject of the following section where the notions of sensors, measurements and output operators will be discussed for the observation of cellular automata by a mobile sensor network.

2.3.2 Sensors, Measurements and Output Operator

Sensors play an important role in the observation of systems. It is through the information provided by them that the state of the system can be reconstructed. In all this work, we consider sensors as mathematical objects capable of measuring a part of the system. The reality is quite different, sensors are often very complex parts and require work in different disciplines, both in physics to convert the measured value into an electrical signal, and in signal processing to amplify, quantify, and filter the measurement. In addition, the sensor has to be adapted to the application: a temperature sensor for a freezer is not the same as one that measures the temperature of the Sun. For this reason, we will consider sensors only as mathematical entities measuring the $s_t(c)$ state of a cell. For example, in the case of the Karafyllidis and Thanailakis forest fire spread model, the cell state is the biomass consumed. This value could be obtained from temperature sensors placed on trees, from satellite imagery of infrared radiation or from any other method. We will consider all these methods as one sensor, the sensor that measures the consumed biomass.

Before discussing measurements we need to define the notion of sensors, in particular its position (*i.e.* the position of these measurements) in relation to the CA model of the observed system. A sensor can measure the state of several cells at once, as shown in Figure 2.10, therefore we define $\mathcal{L}_{q_i} \subset \mathcal{L}$ as the set of cells measured by the sensor q_i. Moreover, several sensors can measure the system at the same time,

so we define the set of all measured cells $\mathcal{L}_q$ is defined by the union of the sets of measured cells of all sensors as presented in definition 2.3.

Definition 2.3 (Sensors)**.** We denote q_i, $i \in [\![1; Q]\!]$, the sensor measuring the states of a set of cell $\mathcal{L}_{q_i}$ and q, the set of Q sensors that observes the set of cells $\mathcal{L}_q \subset \mathcal{L}$ with:

$$\mathcal{L}_q = \bigcup_{i=1}^{Q} \mathcal{L}_{q_i} \tag{2.20}$$

On several occasions we will consider mobile sensors, that is sensors which have the ability to move or to change their measurement area. In this case, the set of cells observed by a mobile sensor evolves in time. We adapt then definition 2.3 by adding the time dependence. $\mathcal{L}_{q_{i,t}}$ will denote the set of cells measured by the sensor q_i at time t, while $\mathcal{L}_{q,t}$ denotes the set of all cells, measured by all the sensors, at time t.

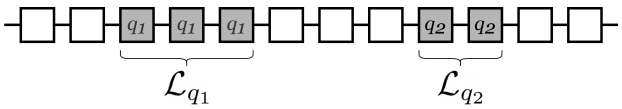


Figure 2.10. Two sensors measuring a one-dimensional CA. Grey cells are measured by a sensor, either q_1 or q_2.

When using a sensor network, it can be considered that the network is dynamic. Sensors may have been added to the network while others may be defective and no longer being able to take measurements. In general, it must be possible to consider that sensors can enter and leave the network, which has an impact on the measured cells. In this case, the set of sensors q can evolve over time (if sensors enter or leave the network). However, it can also be considered that a sensor of the network can be inactive and in this case the set of sensors q does not change but the set of observed cells does. Therefore, the expression of the set of observed cells is defined by :

$$\mathcal{L}_{q,t} = \bigcup_{i=1}^{Q} (\lambda_{i,t} \diamond \mathcal{L}_{q_{i,t}}) \tag{2.21}$$

where $\lambda_{i,t} \in \{0, 1\}$ equal 0 if the sensor q_i is inactive at time t and 1 if it's active. The operator $\diamond$ is defined by:

$$\begin{aligned} \diamond &: \{0,1\} \times \mathcal{P}(\mathcal{L}) \to \mathcal{P}(\mathcal{L}) \\ (\lambda_i, \mathcal{L}_{q_i}) &\mapsto \lambda_i \diamond \mathcal{L}_{q_i} := \begin{cases} \emptyset & \text{if } \lambda_i = 0 \\ \mathcal{L}_{q_i} & \text{else} \end{cases} \end{aligned} \tag{2.22}$$

At each time step, sensors take measurements on all the measured cells and return their state. These measurements are represented by the output vector θ_t which, in

the same way as the s_t configuration, describes the state of the measured cells at time t. We define the sensor measurement operation on the system by the output operator H_t which constructs the measurements θ_t from the measured system state s_t.

Definition 2.4 (Output Operator)**.** The *output operator* H_t maps, at each time t, the state s_t of the cellular automaton $\mathcal{A}$ to the *output vector* $\theta_t \in O = \mathcal{S}^{\mathcal{L}_{q_t}}$:

$$\begin{aligned} H_t &: \mathcal{S}^{\mathcal{L}} \to O \\ s_t &\mapsto \theta_t := s_t(\mathcal{L}_{q,t}) \end{aligned} \tag{2.23}$$

where $s_t(\mathcal{L}_{q,t})$ denotes the state of all cells in set $\mathcal{L}_{q,t}$ at time t.

Remark 2.2. In some cases, the sensors do not measure the state of a particular cell but a value from several cells. For instance, in the example on the CA RNG (see section 5.4 on page 129), the sensor measures the sum modulo 2 of all the cells in the CA. Only one value is returned by the sensor but it depends on all cells. Therefore, definition 2.4 cannot be applied and we must generalise H as:

$$\begin{aligned} H_t &: \mathcal{S}^{\mathcal{L}} \to O \\ s_t &\mapsto \theta_t := h\left(s_t(\mathcal{L}_{q,t})\right) \end{aligned} \tag{2.24}$$

where h is the function that converts the cell state into a θ_t measure.

In general, when observing a system, making a single measurement is not enough to know the state of the system. Measurements must be taken regularly, with a given time step (sampling) and only after a certain period of time, the number of measurements becomes sufficient to determine the current or initial state of the system. The vector composed of the successive output vectors (measurements) generated by the output operator H_t is called an *output sequence*. The output sequence spanning from t_0 to t_T is noted $\Theta_{0,T} = (\theta_0, \theta_1, \ldots, \theta_{T-1})$.

For the analysis of observability, whose criterion will be defined in the next section, we will need to study all possible output sequences. In this case, we derive from the state equation (2.19) the application Θ_T which constructs the output sequence $\Theta_{0,T}$ from the initial configuration s_0 by :

$$\Theta_T : s_0 \mapsto \Theta_{0,T} = (H_0(s_0), H_1 \circ F(s_0), \ldots, H_{T-1} \circ F^{T-1}(s_0)) \tag{2.25}$$

Similarly to what has been done for definition 2.1 of CA, we define the sensor network $\mathcal{H}$ which represents the set of sensors q, the measured cells $\mathcal{L}_q$ and the output operator H_t.

2.3.3 Definition of Observability and Reconstructiblity

Observability and reconstructibility, as defined by Kalman (1963), determine whether or not it is possible to reconstruct the state of a system based on the measurements obtained from one or more sensors. Observability focuses on reconstructing the *initial* state of the system whereas reconstructibility focuses on reconstructing the *current* state of the system. In the case of discrete-time systems, both of these are not equivalent. With a deterministic system, observability is a more general concept since knowing the initial state implies knowing all the evolution of the system. Reconstructibility, on the other hand, is less general but can be easier to assess (as shown in section 5.3 on page 122).

An initial configuration s_0 is considered observable if from the measurements taken by the sensors (*i.e.* the output sequence $\Theta_T(s_0)$) it is possible to uniquely identify s_0. In other words, the output sequence $\Theta_T(s_0)$ must be unique to s_0. Moreover, if all configurations of the CA are observable then the CA is considered observable. Formally, we can define the observability of a CA $\mathcal{A}$ as follows.

Definition 2.5 (Observability)**.** A cellular automaton $\mathcal{A}$ is **observable** by a sensor network $\mathcal{H}$ at time T if and only if:

$$\forall s_0', s_0'' \in \mathcal{S}^{\mathcal{L}}, \Theta_T(s_0') = \Theta_T(s_0'') \implies s_0' = s_0'' \tag{2.26}$$

This definition of observability is equivalent to the injectivity of the output sequence Θ_T. Therefore, proving that a CA is observable by sensors is equivalent to proving the injectivity of the output sequence.

Reconstructibility can similarly be defined as the ability to reconstruct the current state $s_T = F^T(s_0)$ from the output sequence $\Theta_T(s_0)$. However the formulation of definition 2.6 is not the injectivity of Θ_T but the injectivity of Θ_T **with respect to**[3] F^T.

Definition 2.6 (Reconstructiblity)**.** A cellular automaton $\mathcal{A}$ is **reconstructible** by a sensor network $\mathcal{H}$ at time T if and only if:

$$\forall s_0', s_0'' \in \mathcal{S}^{\mathcal{L}}, \Theta_T(s_0') = \Theta_T(s_0'') \implies F^T(s_0') = F^T(s_0'') \tag{2.27}$$

These two definitions are the basis for observation as they ensure that the initial or current state can be reconstructed. It will be necessary to check before the observation phase that the chosen sensor network (by considering the positions or trajectories of each sensor) allows observability and reconstructibility.

[3] The notion of compared injectivity is presented in appendix A. A definition and properties, including a characterisation for the linear case, are given there.

Remark 2.3. In the case where the CA is reversible, then observability and reconstructibility are equivalent notions. Indeed, with reversibility it is possible to find s_0 from s_T because each configuration has a unique antecedent by F. This also results from proposition A.3 on page A-3 and the injectivity (reversibility) of F (and by extension F^T.).

While studying the algorithmic complexity of computing observability of Boolean networks (a notion presented in detail in section 3.3.1), Laschov *et al.* (2013) present an observability criterion for an observability graph, an augmentation of the state transition diagram with the measurements made by the sensor. To be observable, the graph must satisfy two conditions: no configuration has two predecessor configurations with the same output; all cycles generate different output sequences. These two conditions are presented in detail in section 3.3.1 but we will focus here on the first one which can be formally written as:

Definition 2.7 (Adaptability (or Laschov property))**.** A sensor network $\mathcal{H}$ is adapted for the observation of a CA $\mathcal{A}$ if and only if:

$$A\colon \forall s_0', s_0'' \in \mathcal{S}^{\mathcal{L}} \text{ and } s_0' \neq s_0'', F(s_0') = F(s_0'') \implies H(s_0') \neq H(s_0'') \tag{2.28}$$

Laschov *et al* presented this condition as a criterion of observability but it turns out that this condition is not necessary for reconstructibility (see proposition 2.1). Therefore, we generalise this condition as a property of a sensor network with respect to a CA. This condition makes it possible to check before the observability computation that a sensor network $\mathcal{H}$ is adapted to the observation of a CA $\mathcal{A}$. In some cases, this property can even be used to condition the placement of sensors.

Proposition 2.1. *Let $\mathcal{A}$ a cellular automaton and $\mathcal{H}$ a sensor network. The following assertions are true:*

(a) *If $\mathcal{H}$ is not adapted to the observation of $\mathcal{A}$ then $\mathcal{A}$ is **not observable** by $\mathcal{H}$ for any $T > 0$.*

(b) *If $\mathcal{H}$ is **adapted** to the observation of $\mathcal{A}$ and $\mathcal{A}$ is **not observable** by $\mathcal{H}$ for $T > 0$ then $\mathcal{A}$ is **not reconstructible** by $\mathcal{H}$ at time T.*

Proof. Proof of (a): Suppose that $\mathcal{H}$ is **not adapted** to the observation of $\mathcal{A}$, then there exists $s_0', s_0'' \in \mathcal{S}^{\mathcal{L}}$ and $s_0' \neq s_0''$ such that $F(s_0') = F(s_0'') \wedge H(s_0') = H(s_0'')$. Let $s_1 = F(s_0') = F(s_0'')$, we have

$$\begin{aligned}
&\implies H(s_0') = H(s_0'') \wedge \forall k > 1, H(F^k(s_0')) = H(F^k(s_0')) = H(F^{k-1}(s_1)) \\
&\implies \forall T > 0, \Theta_T(s_0') = \Theta_T(s_0'') \\
&\implies \text{For any } T > 0, \mathcal{A} \text{ is not observable by } \mathcal{H}
\end{aligned}$$

Proof of (b): Suppose that $\mathcal{A}$ is not observable by $\mathcal{H}$ for $T > 0$, then there exists $s_0', s_0'' \in \mathcal{S}^{\mathcal{L}}$ such that $\Theta_T(s_0') = \Theta_T(s_0'') \wedge s_0' \neq s_0''$. Then:

$$\forall 0 \leq t < T, H(F^t(s_0')) = H(F^t(s_0'')) \wedge s_0' \neq s_0''$$

However, with the contraposition of (2.28) we have:

$$\forall s_0', s_0'' \in \mathcal{S}^{\mathcal{L}} \text{ and } s_0' \neq s_0'', H(s_0') = H(s_0'') \implies F(s_0') \neq F(s_0'')$$

Hence:

$$\forall 0 \leq t < T, F_{t+1}(s_0') \neq F_{t+1}(s_0'') \wedge s_0' \neq s_0''$$

Especially for $t = T - 1$, we have $F^T(s_0') \neq F^T(s_0'')$ so:

$$\Theta_T(s_0') = \Theta_T(s_0'') \wedge F^T(s_0') \neq F^T(s_0'')$$

This proves that $\mathcal{A}$ is not reconstructible by $\mathcal{H}$ at time T □

The proposition 2.1 guarantees, before the calculation of the output sequence, whether or not the sensor network allows observability. When the proposition allows it but observability is not ensured, then it is unnecessary to evaluate the reconstructibility. Therefore, we only evaluate reconstructibility when the adaptability property does not hold. Moreover, when the CA is reversible, *i.e.* F is injective (there is no s_0', s_0'' such that $s_0' \neq s_0'' \wedge F(s_0') = F(s_0'')$), the property A is always satisfied and thus all sensor networks are adapted for observation (but does not necessarily ensure observability).

In the case of a time-dependent sensor network $\mathcal{H}$, definition 2.7 cannot be applied, it should be verified that the output operator is suitable for all times. In that case, we propose the following definition.

Definition 2.8. A time-dependent sensor network $\mathcal{H}$ is adapted for the observation of a CA $\mathcal{A}$ at time t if and only if:

$$A_t : \forall s_t', s_t'' \in F^t(\mathcal{S}^{\mathcal{L}}) \text{ and } s_t' \neq s_t'', F(s_t') = F(s_t'') \implies H_t(s_t') \neq H_t(s_t'') \quad (2.29)$$

Given this property A_t, the output operator H_t is adapted to the observation over a time horizon T if the following property

$$A = A_0 \wedge A_1 \wedge \cdots \wedge A_{T-1} \tag{2.30}$$

holds.

Only property (b) of proposition 2.1 holds in the case of a time-dependent sensor network. Indeed, the property (b) is easily proved because if we have A then A_t is true for all $t < T$. However, property (a) only holds when $\neg A_0$. If we assume $A_0 \wedge \neg A_1$ then $\forall i \geq 1, H(s_i') = H(s_i'')$ (see the proof (a) of proposition 2.1) although no evidence is present for $H_0(s_0') = H_0(s_0'')$, which might be sufficient to guarantee observability. Therefore we define proposition 2.2 by:

Proposition 2.2. *Let $\mathcal{A}$ a cellular automaton and $\mathcal{H}$ a time-dependent sensor network. The following assertions are true:*

*(a) If $\mathcal{H}$ is not adapted to the observation of $\mathcal{A}$ at time $t = 0$ then $\mathcal{A}$ is **not observable** by $\mathcal{H}$ for any $T > 0$.*

*(b) If $\mathcal{H}$ is **adapted** to the observation of $\mathcal{A}$ for any time $0 \geq t < T$ and $\mathcal{A}$ is **not observable** by $\mathcal{H}$ for $T > 0$ then $\mathcal{A}$ is **not reconstructible** by $\mathcal{H}$ at time T.*

2.4 Conclusion

Throughout this second chapter, we have laid the foundations for the observation of cellular automata (CA) that will be used in the following chapters. In the first part of this chapter, we presented in detail the definition of CA from El Yacoubi. From this definition, we have presented several families of CA that will be used in chapters 3 to 5 for the presentation of different observation criteria. Among these, we will study in particular the additive (or affine) cellular automata (ACA) at the beginning of chapter 3 and the elementary cellular automata (ECA) throughout chapter 4.

In the second part of this chapter, we have formally defined all the tools necessary for the observation of CA. First, we defined the notions of sensors and the output operator, which allows the use of a sensor network, containing both static and mobile sensors. Then, we defined the notions of observability, reconstructibility and adaptability for CA. Observability and reconstructibility are concepts from "classical" control theory that have been adapted to CA. Whereas adaptability is a notion defined by Laschov *et al.* (2013) as a criterion for observability of Boolean networks (BN), it has been translated as specific notion for the observation of CA.

The main contributions of this chapter are the definition of tools for the verification of observability and reconstructibility and the generalisation of the Laschov

criterion by defining adaptability as an observation tool. The definition of observability and reconstructibility have been presented in several publications (Plénet *et al.*, 2020a, 2022a). The discussion on adaptability has not been published yet.

In the following chapters, we will use the definitions made in this chapter to provide methods for observing CA. The criteria of observability and reconstructibility are studied in chapter 3. The chapter 4 focuses on state estimation through CA synchronisation, without checking for observability and reconstructibility.

Chapter 3

Observability and Reconstructibility

Résumé

Dans le chapitre précédent, nous avons défini les automates cellulaires, ainsi que des notions d'observabilité et de reconstructibilité qui s'appliquent à ceux-ci. Dans ce chapitre, nous allons donner des critères qui permettent de vérifier à la fois l'observabilité et la reconstructibilité pour certaines familles d'automates cellulaires.
Dans un premier temps, nous étendrons le critère d'observabilité classique de Kalman aux automates cellulaires additifs. Nous donnerons aussi un critère de reconstructibilité pour cette famille. Ensuite, nous utiliserons la représentation par graphe de transition d'état, ainsi que des résultats de l'observabilité et de la reconstructibilité des réseaux booléens, pour proposer un algorithme permettant de vérifier l'observabilité et la reconstructibilité des automates cellulaires non linéaires (non additifs). Finalement, nous donnerons un critère d'observabilité régionale et de décentralisation de l'observabilité pour les automates cellulaires non linéaires.

Contents

3.1 Introduction

In the previous chapter, we presented the definition for observability, reconstructibility and adaptability. However, we did not provide any useful method to verify those properties. In this chapter we present different methods to check verify those three criteria. Firstly, for ACA and secondly for non-linear CA.

ACA are linear systems with a finite number of variables. We can therefore apply some results from the control theory of linear time invariant (LTI) systems. However, we must be careful as we are dealing with finite-state variables. We present the Kalman rank condition for observability of ACA. We present a similar condition for reconstructibility and adaptability.

In the second part, we study non-linear CA by making them Boolean networks. Therefore, we adapt the work made on observability and reconstructibility of Boolean network to non-linear CA. The computational complexity of this method makes it impossible to apply it on large CA (more than a few cells). We present two different approaches to reduce the complexity. The first use some condition on the initial configuration and the second uses a decentralisation of the observability or reconstructibility.

3.2 Kalman Criterion for Additive and Affine Cellular Automata

The Kalman (1963) rank condition (or Kalman criterion) is a necessary and sufficient condition for a linear system to be controllable. Due to the duality of control and observation, this condition also applies to observability. The obtained observability condition is then based on the full rank of the observability matrix, which ensures the injectivity of the output sequence. Initially defined for linear time invariant systems, many other types of systems such as discrete-time systems (Sarachik and Kreindler, 1965) have a Kalman condition for controllability and observability. In this section, we propose an extension of the Kalman observability condition for ACA, and corresponding criteria for reconstructibility and adaptability. Finally, we present numerical methods for the verification of observability and reconstructibility for CA with a large number of cells.

3.2.1 *Affine and Additive State Representation*

In order to apply the Kalman criterion, we need a linear state space representation similar to the one presented in equation (2.16). In addition, we also need to be able to express the output sequence (2.25) with a matrix representation in order to

verify its injectivity. We can extend the ACA to an affine sensor network when the corresponding output operator is an affine map. Such an affine output operator can be written as

$$H\colon \mathcal{S}^N \to \mathcal{S}^Q$$

$$x_t \mapsto y_t = Cx_t + \gamma = \begin{bmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,N} \\ c_{2,1} & c_{2,2} & \dots & c_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ c_{Q,1} & c_{Q,2} & \dots & c_{Q,N} \end{bmatrix} \cdot \begin{bmatrix} s_t(c_1) \\ s_t(c_2) \\ \vdots \\ s_t(c_N) \end{bmatrix} + \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_Q \end{bmatrix} \tag{3.1}$$

Note that, in many scenarios, sensors directly measure the state of a cell and therefore the weights $c_{i,j}$ of the matrix C are ones and the constants γ_i are zeros. Using the global transition function of an ACA (2.12), together with the affine output operator (3.1), the state-space representation of the ACA extended with its affine sensor network reads:

$$\begin{cases} x_{t+1} &= Ax_t + \eta \\ y_t &= Cx_t + \gamma \end{cases} \tag{3.2}$$

We can express both x_t and y_t as a function of the initial configuration x_0:

$$\begin{cases} x_t &= A^t x_0 + J_{t-1}\eta \\ y_t &= C_t A^t x_0 + C_t J_{t-1}\eta + \gamma \end{cases} \tag{3.3}$$

with $J_t = \sum_{k=0}^{t} A^k$

Finally, by adapting the definition of the output sequence (2.25) with the previous formulation, we can give the matrix writing for the output sequence Θ_T.

Definition 3.1 (Affine Output Sequence)**.** The output sequence Θ_T can be represented in affine form using the affine forms of the transition function and the output operator. We note Y_T the output sequence for T outputs:

$$Y_T = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{T-1} \end{bmatrix} = \underbrace{\begin{bmatrix} C_0 \\ C_1 A \\ \vdots \\ C_{T-1}A^{T-1} \end{bmatrix}}_{O_T} x_0 + \underbrace{\begin{bmatrix} \gamma_0 \\ C_1 J_0 \eta + \gamma_1 \\ \vdots \\ C_{T-1} J_{T-2}\eta + \gamma_{T-1} \end{bmatrix}}_{\Gamma_T} = O_T x_0 + \Gamma_T \tag{3.4}$$

where O_T is called the observability matrix and Γ_T is a constant vector. The linear map represented by the observability matrix O_T carries the injectivity property of the output sequence and therefore the observability property.

Example 3.1

In order to illustrate the results given in this section, we will apply them in a simple and didactic example where the necessary calculations will be detailed. In this first example, we will define the CA and the sensor networks which will also be used in examples 3.2 and 3.3 for the verification of observability and reconstructibility. In order to be able to detail the calculations, the A and O matrices, whose sizes depend on the number of cells in the CA, must remain "readable". Therefore we decided to use the one-dimensional ACA with 8 cells defined by:

- $\mathcal{L} = \{0, 1, \ldots, 7\}$, a one-dimensional lattice of 8 square cells.
- $\mathcal{S} = \{0, 1, 2, 3, 4\}$, a set of $k = 5$ states. We have chosen k as a prime number in order to apply observability and reconstructibility.
- $\mathcal{N}(c_i) = \{c_{i-1}, c_i, c_{i+1}\}$, the neighbourhood of cell c_i, with the periodic boundaries conditions $c_{-1} = c_7$ and $c_8 = c_0$.
- $f\colon s_t(\mathcal{N}(c_i)) \mapsto s_{t+1}(c_i) = 2 \cdot s_t(c_{i-1}) + 3 \cdot s_t(c_i) + 1 \cdot s_t(c_{i+1}) + 4$, an affine transition function

In order to obtain the matrix form of the CA evolution, we will use the definition of the matrix A from (2.11). The weight $a_{i,j}$ are derived from the weight of the local expression of f:

$$\forall i, j \in \mathcal{L}, a_{i,j} = \begin{cases} 3 & \text{if } j = i \\ 2 & \text{if } j \equiv i - 1 \mod n \\ 1 & \text{if } j \equiv i + 1 \mod n \\ 0 & \text{else} \end{cases}$$

Therefore, we obtain the following matrix formulation:

$$x_{t+1} = Ax_t + \eta = \begin{bmatrix} s_{t+1}(c_0) \\ s_{t+1}(c_1) \\ s_{t+1}(c_2) \\ s_{t+1}(c_3) \\ s_{t+1}(c_4) \\ s_{t+1}(c_5) \\ s_{t+1}(c_6) \\ s_{t+1}(c_7) \end{bmatrix} = \begin{bmatrix} 3 & 1 & 0 & 0 & 0 & 0 & 0 & 2 \\ 2 & 3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 3 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 2 & 3 \end{bmatrix} \cdot \begin{bmatrix} s_t(c_0) \\ s_t(c_1) \\ s_t(c_2) \\ s_t(c_3) \\ s_t(c_4) \\ s_t(c_5) \\ s_t(c_6) \\ s_t(c_7) \end{bmatrix} + \begin{bmatrix} 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \end{bmatrix}$$

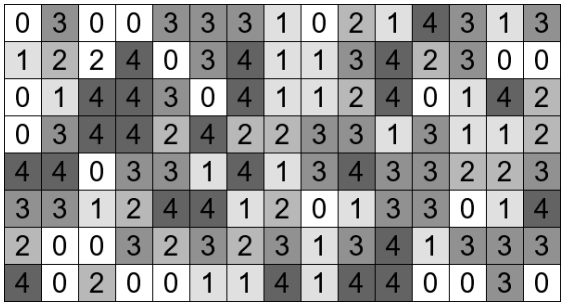


FIGURE 3.1. Example of evolution of the ACA from the initial configuration 01004324 for 15 time steps. Time is going rightward.

In order to illustrate the use of both mobile and static sensors, we will define $\mathcal{H}^{(1)}$ and $\mathcal{H}^{(2)}$, two sensor networks. $\mathcal{H}^{(1)}$ describes two static sensor that measures the left-end and right-end cells. $\mathcal{H}^{(2)}$ describes a mobile sensor that observes the left-end cell but moves to the right cell (with a loop due to the periodic boundaries) at every time steps. The measurement sets of $\mathcal{H}^{(1)}$ and $\mathcal{H}^{(2)}$ are denoted respectively $Lc_q^{(1)}$ and $Lc_q^{(2)}$:

- $\mathcal{L}_q^{(1)} = \{c_0, c_7\}$
- $\mathcal{L}_{q,t}^{(2)} = \{c_i | \text{ with } i = t \mod 5\}$

Both sensor networks directly measure the state of the cells. Thus both $H^{(1)}$ and $H_t^{(2)}$ are additive output operators which satisfy definition 2.4. Their corresponding C matrices can be written respectively as:

- $H^{(1)}$: $C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$
- $H_t^{(2)}$:
  - $C_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$
  - $C_1 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$
  - …
  - $C_7 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

In this first example, we presented a one-dimensional ACA as well as two sensor networks, one of which is time-dependent. Later, we will study observability (example 3.2) as well as reconstructibility (example 3.3) for those sensor networks.

3.2.2 Conditions for Observability and Reconstructibility

In this section, we present theorems 3.1 and 3.3, both derived from the Kalman condition of classical control theory, which provide characterisations respectively for observability and reconstructibility. Each of these two theorems is accompanied by a corollary that allows either the reconstruction of the initial or the current state when respectively observability or reconstructibility is satisfied.

Theorem 3.1 (Kalman Condition). *Let $\mathcal{A}$ and $\mathcal{H}$ be an ACA and time-dependent sensor network, with the associated matrices respectively denoted A and C_t.*

*The ACA $\mathcal{A}$ is **observable** by the sensor network $\mathcal{H}$ if and only if there exists a time $T > 0$ such that:*

$$rank\ O_T = \begin{bmatrix} C_0 \\ C_1 A \\ \vdots \\ C_{T-1} A^{T-1} \end{bmatrix} = N \tag{3.5}$$

The following proof demonstrates theorem 3.1 for an affine CA and an affine sensor network. For additive CA or sensor network, the proof is essentially the same with the only difference that the constants γ and η are zero vectors.

Proof. Let $\mathcal{A}$ be an affine CA with A and η its associated matrix and constant vector. Let $\mathcal{H}$ be a time-dependent affine sensor network associated with the matrix C_t and the constant vector γ_t. Then, let $x_0 \in \mathcal{S}^N$ be the initial configuration and $Y_T = \begin{bmatrix} y_0 & y_1 & \dots & y_{T-1} \end{bmatrix}$ the output sequence generated by the sensor network such that $Y_T = O_T x_0 + \Gamma_T$.

Considering the definition 2.5 of observability, then:

$$\begin{aligned}
\mathcal{A} \text{ is observable by } \mathcal{H} &\iff \forall s_0', s_0'' \in \mathcal{S}^{\mathcal{L}}, \Theta_T(s_0') = \Theta_T(s_0'') \implies s_0' = s_0'' \\
&\iff \forall x_0', x_0'' \in \mathcal{S}^N, O_T x_0' + \Gamma_T = O_T x_0'' + \Gamma_T \implies x_0' = x_0'' \\
&\iff \forall x_0', x_0'' \in \mathcal{S}^N, O_T(x_0' - x_0'') = 0 \implies (x_0' - x_0'') = 0 \\
&\iff \ker\ O_T = \{0\} \\
&\iff rank\ O_T = \dim(S^N) - \dim(\ker\ O_T) = N
\end{aligned}$$

□

In the general case, the number of measurements (*i.e.* the dimension of the output sequence) is not necessarily equal to the number of cells. As a result, the matrix O_T is not a square matrix. It is therefore impossible to find the inverse of the observability

matrix even if O_T is full rank. There is, however, a pseudo-inverse (Ben-Israel and Greville, 2003, Chap 1) matrix $O_T^\dagger$ which acts as a left inverse such that $O_T^\dagger O_T = I_N$. Using this pseudo-inverse, it is possible to state the corollary 3.1.

Corollary 3.1. *If the Kalman criterion is satisfied, then it is possible to reconstruct the initial state by pseudo-inverting the observability matrix. Indeed, based on the formulation (3.4) we obtain*

$$x_0 = O_T^\dagger (Y_T - \Gamma_T) \tag{3.6}$$

Proof. Consider an ACA $\mathcal{A}$ observable by an affine sensor network $\mathcal{H}$ such that $\forall x_0 \in \mathcal{S}^N$, $Y_T = O_T x_0 + \Gamma_T$ and $rank\ O_T = N$. To simplify the notations, we shall simply note O and Γ to respectively represent O_T and Γ_T.

As $rank\ O = N$, it means that it exists P such that $PO = I$ (but not necessarily $OP = I$ because O is full column rank not full-row rank). We can find $P = O^\dagger$ by computing the pseudo-inverse of O. Using the equation (3.4) we find that:

$$Y_T = Ox_0 + \Gamma \iff O^\dagger (Y_T - \Gamma) = O^\dagger O x_0 \iff O^\dagger (Y_T - \Gamma) = x_0 \tag{3.7}$$

If O is full column rank, $O^\dagger = (O'O)^{-1}O'$. If O is square then $O^\dagger = O^{-1}$. □

A characterisation theorem, similar to theorem 3.1, may also be formulated for the adaptability property. Recall that a sensor network is said adapted for the observation of a CA when no configuration has two predecessors with the same output (see definition 2.7). In the case of ACA, the adaptability characterisation theorem may be formulated as follows.

Theorem 3.2. *Let $\mathcal{A}$ and $\mathcal{H}$ be an ACA and affine sensor network, with associated matrices respectively denoted A and C. The sensor network $\mathcal{H}$ is **adapted to the observation** of the ACA $\mathcal{A}$ if and only if :*

$$\ker A \cap \ker C = \{0\} \tag{3.8}$$

Proof. Let $\mathcal{A}$ an ACA and A and η its associated matrix and constant vector. Let $\mathcal{H}$ be a sensor network associated with the matrix C and the constant vector γ. The adaptability condition (2.28) in definition 2.7, when applied to the state space representation (3.2) for ACA, reads:

$$\begin{aligned}
&\forall x_0', x_0'' \in \mathcal{S}^N \wedge x_0' \neq x_0'', Ax_0' + \eta = Ax_0'' + \eta \implies Cx_0' + \gamma \neq Cx_0'' + \gamma \\
&\iff \forall x_0', x_0'' \in \mathcal{S}^N \wedge x_0' \neq x_0'', A(x_0' - x_0'') = 0 \implies C(x_0' - x_0'') \neq 0 \\
&\iff \forall x_0 \in S^N \wedge x_0 \neq 0, x_0 \in \ker A \implies x_0 \notin \ker C \\
&\iff \ker A \cap \ker C = \{0\}
\end{aligned}$$

□

Using theorem 3.2, it is possible to find conditions on the C matrix of the output operator by carefully placing the sensors so that it respects the theorem. In example 3.2, in addition to checking whether or not the sensor networks $\mathcal{H}^{(1)}$ and $\mathcal{H}^{(2)}$ are adapted to the observation, an example is given to provide a condition on the position of the sensor for the sensor network to be adapted to the observation.

Example 3.2

In this example, we take the ACA and the two sensor networks of example 3.1 and we will check the observability. If it is validated, then we will reconstruct the initial state using the output sequences generated by the output operators on the execution of Figure 3.1.

Before assessing the observability, we must verify if the sensors are adapted to the observation of the ACA. Therefore we find the eigenvectors of ker A which are

$$v_A^{(1)} = \begin{bmatrix} 2 \\ 2 \\ 0 \\ 1 \\ 2 \\ 2 \\ 0 \\ 1 \end{bmatrix} ; v_A^{(2)} = \begin{bmatrix} 3 \\ 1 \\ 1 \\ 0 \\ 3 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Considering e_i, the vectors of the canonical basis of S^N, we can write the vectors $V_A^{(1)}$ and $V_A^{(2)}$ as:

$$\begin{aligned} V_A^{(1)} &= 2e_0 + 2e_1 + e_3 + 2e_4 + 2e_5 + e_7 \\ V_A^{(2)} &= 3e_0 + e_1 + e_2 + 3e_4 + e_5 + e_6 \end{aligned}$$

Thus every element x of ker A can be written in the form $\alpha V_A^{(1)} + \beta V_A^{(2)}$ with $(\alpha; \beta) \in \mathcal{S} \times \mathcal{S}$. In the canonical basis, we obtain

$$\begin{aligned} x &= \alpha V_A^{(1)} + \beta V_A^{(2)} \\ &= (2\alpha + 3\beta)e_0 + (2\alpha + \beta)e_1 + \beta e_2 + \alpha e_3 + \dots \\ &\cdots + (2\alpha + 3\beta)e_4 + (2\alpha + \beta)e_5 + \beta e_6 + \alpha e_7 \end{aligned}$$

When a sensor directly measures the state of a cell, all these measurements are only sensitive to changes in that cell. Thus, the kernel of the output operator is composed of all the other cells. The eigenvectors of $\ker H^{(1)}$ are therefore e_1, e_2, e_3, e_4, e_5 and e_6. For $H_t^{(2)}$, these are all e_i so that $i \neq t$.

To guarantee the adaptability of the sensor network, it is necessary to find $\ker A \cap \ker C = \{0\}$, thus it is necessary to be able to nullify the coefficients in front of the e_i in the writing of x for those which are not $\ker C$, *i.e.* those corresponding to the measured cells. If there exists some values of α and β, for which $\alpha V_A^{(1)} + \beta V_A^{(2)} \in \ker H$ and $(\alpha; \beta) \neq (0; 0)$, then $\ker A \cap \ker C \neq \{0\}$ the adaptability is not respected.

Finally, we can see that $\mathcal{H}^{(2)}$ is not adapted to the observation whatever t is, because there exists non-zero solutions for each of the vectors: $(1, 1)$ is a solution of $2\alpha + 3\beta = 0$; $(1, 3)$, $(2, 1)$, $(3, 4)$ and $(4, 2)$ are solutions of $2\alpha + \beta = 0$; $(0, \beta \neq 0)$ is a solution of $\alpha = 0$ and $(\alpha \neq 0, 0)$ is a solution of $\beta = 0$. However, $\mathcal{H}^{(1)}$ is adapted as the non-zero solutions are not compatible, $(1, 1)$ is not solution of $\alpha = 0$. More generally, any output operator that observes a combination of at least two cells c_i and c_j, with $i \not\equiv j \mod 4$, is suitable for the observation of this ACA.

Even if the sensor network $\mathcal{H}^{(2)}$ cannot ensure observability, we will still compute the observability matrix (this will be useful when computing reconstructibility) and check its rank. First, we must decide on a time horizon T in order to construct the matrix O_T. Let $T = 4$ (this is justified by the Cayley-Hamilton Theorem presented later on) for the static sensor and $T = 8$ for the mobile one, we obtain the following expression for Y_T in the case of $\mathcal{H}^{(1)}$ and $\mathcal{H}^{(2)}$:

$$Y_T^{(1)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 3 & 1 & 0 & 0 & 0 & 0 & 0 & 2 \\ 1 & 0 & 0 & 0 & 0 & 0 & 2 & 3 \\ 3 & 1 & 1 & 0 & 0 & 0 & 4 & 2 \\ 1 & 1 & 0 & 0 & 0 & 4 & 2 & 3 \\ 3 & 3 & 4 & 1 & 0 & 3 & 1 & 1 \\ 3 & 4 & 1 & 0 & 3 & 1 & 1 & 3 \end{bmatrix} \cdot x_0 + \begin{bmatrix} 0 \\ 0 \\ 4 \\ 4 \\ 3 \\ 3 \\ 2 \\ 2 \end{bmatrix}$$

$$Y_T^{(2)} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 2 & 3 & 1 & 1 & 0 & 0 & 0 \\ 3 & 1 & 1 & 3 & 3 & 4 & 1 & 0 \\ 2 & 1 & 3 & 0 & 1 & 0 & 2 & 2 \\ 2 & 0 & 1 & 0 & 0 & 3 & 0 & 0 \\ 4 & 1 & 4 & 3 & 1 & 1 & 4 & 3 \\ 1 & 2 & 0 & 4 & 0 & 3 & 2 & 4 \end{bmatrix} \cdot x_0 + \begin{bmatrix} 0 \\ 4 \\ 3 \\ 2 \\ 1 \\ 0 \\ 4 \\ 3 \end{bmatrix}$$

In the case of $\mathcal{H}^{(2)}$, the rank of the observability matrix is 7 which is not enough to ensure observability. The verification of the reconstructibility will be done in example 3.3. The computation of the rank must be done in modular arithmetic, which is not the case for the usual rank operators. Indeed, the observability matrix for $\mathcal{H}^{(2)}$ is not full rank, although using the usual rank operator it would be full rank since its determinant is 5 (which is 0 in modular arithmetic modulo 5).

On the contrary, $\mathcal{H}^{(1)}$ ensures the observability because the rank of its observability matrix is 8. We will now compute the $O^\dagger$ matrix in order to reconstruct the initial configuration.

In this case $O^\dagger$ is a square matrix, so we will use the inverse computed from the determinant and the adjugate matrix (transpose of the cofactor matrix) $O^{-1} = det(O)^{-1} \cdot adj(O)$. Knowing that the determinant is 4, its inverse is 4 because $4 \cdot 4 = 1 \mod 5$. Thus, the inverse of O is expressed by:

$$
\begin{aligned}
O^\dagger &= det(O)^{-1} \cdot adj(O) \\
&= 4 \cdot \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 2 & 4 & 0 & 0 & 0 & 0 & 0 \\ 3 & 4 & 1 & 2 & 4 & 0 & 0 & 0 \\ 0 & 4 & 2 & 3 & 4 & 2 & 4 & 0 \\ 4 & 0 & 3 & 4 & 2 & 3 & 0 & 3 \\ 3 & 2 & 4 & 4 & 0 & 1 & 0 & 0 \\ 3 & 4 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 4 & 3 & 1 & 0 & 0 & 0 \\ 0 & 1 & 3 & 2 & 1 & 3 & 1 & 0 \\ 1 & 0 & 2 & 1 & 3 & 2 & 0 & 2 \\ 2 & 3 & 1 & 1 & 0 & 4 & 0 & 0 \\ 2 & 1 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\end{aligned}
$$

With the corollary 3.1 and the $O^\dagger$ matrix, we are able to reconstruct the initial state x_0 from the output sequence Y_T:

$$
\begin{aligned}
x_0 &= O_T^\dagger (Y_T - \gamma_T) \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 4 & 3 & 1 & 0 & 0 & 0 \\ 0 & 1 & 3 & 2 & 1 & 3 & 1 & 0 \\ 1 & 0 & 2 & 1 & 3 & 2 & 0 & 2 \\ 2 & 3 & 1 & 1 & 0 & 4 & 0 & 0 \\ 2 & 1 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \left(\begin{bmatrix} 0 \\ 4 \\ 3 \\ 0 \\ 0 \\ 2 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 4 \\ 4 \\ 3 \\ 3 \\ 2 \\ 2 \end{bmatrix} \right) = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 4 \\ 3 \\ 2 \\ 4 \end{bmatrix}
\end{aligned}
$$

In this example we were able to verify the observability of the output operator

$\mathcal{H}^{(1)}$ and to reconstruct the initial state. However, the $\mathcal{H}^{(2)}$ output operator does not guarantee observability, so we will check observability in example 3.3.

In the case of time-continuous linear systems, observability and reconstructibility are equivalent concepts (Antoulas, 2005, section 4.2.2). The Kalman criterion ensures observability. However, in discrete time, there are systems which are reconstructible but are not observable (see for instance the road traffic example in section 5.3 on page 122). For this reason we propose a new theorem, similar to the Kalman condition, that provides a necessary and sufficient condition for an ACA to be reconstructible.

Theorem 3.3 (Reconstructibility Criterion)**.** *Let $\mathcal{A}$ and $\mathcal{H}$ be an ACA and affine sensor network. We denote A, C their matrix form. The ACA $\mathcal{A}$ is said* ***reconstructible*** *by the sensor network $\mathcal{H}$ if and only if there exists $T > 0$ such that:*

$$\ker O_T \subset \ker A^T \tag{3.9}$$

Proof. Let $\mathcal{A}$ an ACA and A and η its associated matrix and constant. Let $\mathcal{H}$ be a time-dependent affine sensor network associated to the matrix C_t and the constant γ_t at time t. Then, let $x_0 \in \mathcal{S}^N$ be the initial state and $Y_T = \begin{bmatrix} y_0 & y_1 & \dots & y_{T-1} \end{bmatrix}$ the output sequence generated by the output operator such that $Y_T = O_T x_0 + \Gamma_T$.

Consider the definition 2.6 of the global observability, then:

$$\begin{aligned}
&(\mathcal{A}, \mathcal{H}) \text{ is reconstructible} \\
&\iff \forall s_0', s_0'' \in \mathcal{S}^{\mathcal{L}}, \Theta_T(s_0') = \Theta_T(s_0'') \implies F^T(s_0') = F^T(s_0'') \\
&\iff \forall x_0', x_0'' \in \mathcal{S}^N, O_T x_0' + \Gamma_T = O_T x_0'' + \Gamma_T \implies A^T x_0' + J_{T-1}\eta = A^T x_0'' + J_{T-1}\eta \\
&\iff \forall x_0', x_0'' \in \mathcal{S}^N, O_T(x_0' - x_0'') = 0 \implies A^T(x_0' - x_0'') = 0 \\
&\iff ker\ O_T \subset ker\ A^T
\end{aligned}$$

□

Corollary 3.2, hereafter, provides a reconstruction of the current state x_T from the output sequence and the observability matrix. Example 3.3 describes the process of constructing the matrix R in the case of a one-dimensional ACA. This method can easily be extended to multidimensional ACA.

Corollary 3.2. *If the reconstructibility criterion is verified, then it is possible to find a matrix R such that:*

$$x_T = R(Y_T - \Gamma_T) + J_{T-1}\eta \tag{3.10}$$

And Penrose (1956) defines R as a set of solution depending on w :

$$R' = A^T O_T^{\dagger} + (I - O^{\dagger}O)w \tag{3.11}$$

where vector w is arbitrary and since O is not of full rank, its pseudo-inverse is obtained through the matrix of its singular value decomposition U, Σ, and V as $O^{\dagger} = V\Sigma^{\dagger}U'$.

Proof. Consider an ACA $\mathcal{A}$ (with a matrix A and a constant η) reconstructible by an affine sensor network $\mathcal{H}$ such that $Y_T = O_T x_0 + \Gamma_T$ and $\ker O_T \subset \ker A^T$.

As $\ker O_T \subset \ker A^T$, it means there exists a matrix R such that $A^T = RO_T$. With this property, we can find that:

$$\begin{aligned} Y_T = O_T x_0 + \Gamma_T &\iff R(Y_T - \Gamma_T) = RO_T x_0 \\ &\iff R(Y_T - \Gamma_T) + J_{T-1}\eta = A^T x_0 + J_{T-1}\eta \\ &\iff x_T = R(Y_T - \Gamma_T) + J_{T-1}\eta \end{aligned}$$

□

Example 3.3

In this last example, we will check the reconstructibility of the sensor network $\mathcal{H}^{(2)}$, defined in example 3.1, for the case $T = 8$. We proved observability is not satisfied. To prove reconstructibility, we start by finding the eigenvectors of the kernels of O_T and A^T:

$$v_O^{(1)} = \begin{bmatrix} 0 \\ 3 \\ 1 \\ 1 \\ 0 \\ 3 \\ 1 \\ 1 \end{bmatrix} ; v_A^{(1)} = \begin{bmatrix} 2 \\ 2 \\ 0 \\ 1 \\ 2 \\ 2 \\ 0 \\ 1 \end{bmatrix} ; v_A^{(2)} = \begin{bmatrix} 3 \\ 1 \\ 1 \\ 0 \\ 3 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

We can fairly easily see that $v_O^{(1)} = v_A^{(1)} + v_A^{(2)}$ is a linear combination of the eigenvectors of $\ker A^T$. Thus $\ker O_T \subset \ker A^T$ and $\mathcal{H}^{(2)}$ ensures reconstructibility. We will therefore use corollary 3.2 to reconstruct the current state, namely

x_8. However, we did not find any algorithm to compute the pseudo-inverse or the singular value decomposition using modular arithmetic. Nevertheless, to reconstruct the current state, we could always use a state estimator like the one presented in section 3.3.3.

Finally, through these three examples we were able to verify the observability and reconstructibility for ACA. We were able to show the use of both a mobile sensor and a "network" of two static sensors. In section 5.4 on page 129 we will present an example about random number generator attack, where the use of observability for ACA will find a more concrete and natural application.

3.2.3 Tools for the Verification of the Kalman Criterion

In this section, we present tools that aim to simplify the computations for observability and reconstructibility. The Cayley-Hamilton theorem is used to bound the time horizon in the observability calculation with a static sensor network (only the lower bound applies when using a mobile sensor network). Different tools are presented to assist in the numerical verification of observability and reconstructibility.

Cayley-Hamilton Theorem

In linear algebra, the Cayley-Hamilton theorem states that any square matrix A (of size $n \times n$) over a commutative ring (in our case $\mathcal{S}$) satisfies its characteristic equation, *i.e.* A^n is a linear combination of A^i with $i \in [\![0; n-1]\!]$. It can be coupled with the Kalman Condition to give an upper bound on the time horizon T.

Theorem 3.4 (Cayley-Hamilton Theorem)**.** *Suppose a CA observed by a* ***time-invariant*** *sensor network C* (i.e. *by a static sensor), the observability matrix O_T is of size $N \times Q.T$. For the matrix O_T to be of full column rank, the time horizon T is bounded according to:*

$$\frac{N}{Q} \leq T \leq N \tag{3.12}$$

Proof. Let us prove the two inequalities separately:

Proof of $T \leq N$: The Cayley-Hamilton theorem guarantees that the rank of O_T will not increase beyond T = N as CA^N is a linear combination of the CA^i, for $0 \leq i \leq N$.

Proof of $T \geq N/Q$: For O_T to have a rank of N, it needs at least N rows and columns, thus $Q.T \geq N$. □

Remark 3.1. In the case of a mobile sensor (*i.e.* a time-dependent sensor network), only the lower bound of the theorem 3.4 stands true. The Cayley-Hamilton theorem

does not apply because $C_T A^T$ is a linear combination $C_T + C_T A + \cdots + C_T A^{T-1}$ and not of the type $C_0 + C_1 A + \cdots + C_{T-1} A^{T-1}$, where C_t depends explicitly on the time value t.

The Cayley-Hamilton Theorem is particularly useful to limit the amount of time horizons to consider. In example 3.2, for the first output operator $H^{(1)}$, the Cayley-Hamilton theorem states that T is bounded between 4 and 8. Because of it, if the system was not observable at time $T = 8$ then it would have been meaningless to consider a larger time horizon in the expectation that the additional measurements would provide additional information to guarantee observability.

Numerical Tools

When the number of cells of the CA becomes too large, it is impossible to build manually the matrices and do the computations of observability and reconstructibility. In this section, we propose a method for constructing the A matrix as well as a description of the algorithmic complexity of the observability computation.

In the one-dimensional case, the matrix A is often almost similar to a tridiagonal matrix (or with more diagonals if the neighbourhood is larger). The non-zero values outside the main diagonals are typically close from the angles of the matrix and related to the boundary conditions. Except for the case of reflexive boundary conditions, the matrix A can be described as a Toeplitz matrix (Gray, 2006), where the coefficients $a_{\pm i}$ correspond to the weights used in the sum (2.8)[1].

$$A = \begin{bmatrix} a_0 & a_1 & a_2 & \dots & \dots & a_{n-1} \\ a_{-1} & a_0 & a_1 & \ddots & & \vdots \\ a_{-2} & a_{-1} & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_1 & a_2 \\ \vdots & & \ddots & a_{-1} & a_0 & a_1 \\ a_{-(n-1)} & \dots & \dots & a_{-2} & a_{-1} & a_0 \end{bmatrix} \tag{3.13}$$

The coefficient a_0 corresponds to the weight of the central cell, a_i to the cells on the right and a_{-i} to the cells on the left. In the case of periodic boundaries, the weights are also periodic: $a_{n-1} = a_{-1}$ and $a_{-(n-1)} = a_1$. In example 3.1, $a_0 = 3$, $a_1 = 1$ and $a_{-1} = 2$. For reflexive boundaries, we may consider first a null boundaries CA and add the boundaries afterwards by considering the following matrix (for a neighbourhood of distance d):

[1] When considering a two-dimensional (or higher-dimensional) CA, the matrix A is no longer a Toeplitz matrix

$$A = A_{null} + \begin{bmatrix} d \cdot a_0 & 0 & 0 & \dots & \dots & 0 \\ (d-1) \cdot a_{-1} & 0 & 0 & \ddots & & \vdots \\ (d-2) \cdot a_{-2} & 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 & (d-2) \cdot a_2 \\ \vdots & & \ddots & 0 & 0 & (d-1) \cdot a_1 \\ 0 & \dots & \dots & 0 & 0 & d \cdot a_0 \end{bmatrix} \tag{3.14}$$

In most cases, the state of a cell is observed directly, *i.e.* each element of the output vector y is the state of one of the cells of the CA. Therefore the matrix C has only one non-zero element in each row at the position corresponding to the observed cell.

The rank of a matrix is usually computed by performing a Gaussian elimination in order to find the number of independent columns (or rows) that represent the rank of the matrix. As the operations use modular arithmetic, the Gaussian rank elimination must use modular arithmetic. In example 3.2, the rank of the observability matrix differs whether the rank is done in modular arithmetic or not. Standard Gaussian elimination has an asymptotic algorithmic complexity $O(N^3)$, but more efficient algorithms exist to compute the rank more efficiently. For instance, (Cheung *et al.*, 2013) propose an algorithm to compute the rank of a $N \times M$ matrix with an algorithmic complexity of $O(NM^{\omega-1})$, with $\omega \leq 2.38$. When the Gaussian elimination is applied to the rank of the observability matrix O_T of size $N \times Q.T$, the complexity is therefore $O(QTN^{\omega-1})$.

In this section we have been able to study the observability, reconstructibility and adaptability of affine and additive cellular automata. The Kalman rank condition of classical control theory has been adapted to ACA as an observability condition. The reconstructibility and adaptability conditions are based on the kernel of linear applications. The three examples presented provided an opportunity to detail the computations of the different conditions, but also to present the potential computational difficulties related to modular arithmetic. Finally, we discussed typical matrix representation for typical 1D ACA and matrix algorithm complexity issues for the computation of observability and reconstructibility tests.

Although the presented methods perform well in the context of ACA, few CA can be considered as ACA. Still inspired by the classical linear control theory, other issues could be worth studying for ACA. First, the development of a state estimator could be investigated. It would be necessary to ensure that the observability and reconstructibility criteria allow the convergence of the state estimator, as well as to adapt the estimation algorithms to take into account the modular arithmetic. Second, a linearisation method could be found which would make it possible to analyse the observability/reconstructibility of non-linear CA through corresponding results for

their lineasized ACA. This generalisation of results which exists for the state space representation (the linearisation or indirect approach) would allow the use of the results developed here above for a much larger number of CA systems. However, in this thesis, we have instead chosen to exploit the discrete and finite aspect of CA in order to develop a direct approach for the observability and reconstructibility analysis for non-linear CA.

3.3 Observability and Reconstructibility for Non-Linear Cellular Automata

In the previous part, we have studied the observability and the reconstructibility of ACA. They represent a very small part of the existing CA: for instance, only 16 rules out of the 256 Wolfram's ECA can be considered additive or affine.

In order to study the observability and reconstructibility of non-linear CA (*i.e.* neither additive nor affine), we will not focus on the algebraic expression of the transition function as we have done with ACA. Instead, we will study the links between the configurations through the state transition diagram presented in Figure 2.8.

This method has already been studied many times for the verification of the observability and reconstructibility of Boolean Networks (BN) (Laschov *et al.*, 2013; Cheng *et al.*, 2010). In this section, we adapt these formulations to bounded CA and propose numerical methods to verify their observability and reconstructibility. We will start by briefly defining BN, their links with CA as well as the different observability and reconstructibility criteria that are available in literature for BN.

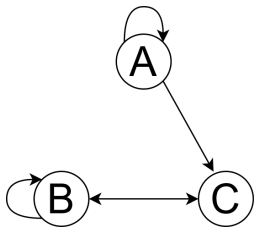
3.3.1 Observability and Reconstructibility of Boolean Network

Described by Kauffman (1969), Random Boolean Networks (RBN) are historically used as a tool for modelling genetic circuits, a new concept on the behaviour of genes inside a cell discovered a few years earlier by Jacob and Monod (winners of the 1965 Nobel Prize in Physiology or Medicine). Since their creation, they have been widely used in biology and particularly for the study of biological systems (Wang *et al.*, 2012). It is only in the early 2000s that they start to be studied mathematically, through the study of fixed points or basins of attraction (Albert and Barabási, 2000). Different variants of BN are appearing, such as Probabilistic Boolean Network (PBN) (Shmulevich *et al.*, 2002) and Boolean Control Networks (BCN) (Cheng and Qi, 2009). The study of BCN by Cheng with the use of the semi-tensor product (Cheng, 2005) introduced Boolean Network to the control community in the 2000s. Since then, BN have been the subject of numerous studies on the controllability (Cheng and Qi, 2009) and the observability (Laschov *et al.*, 2013).

In this section, we will present BN and the relation with CA. We will also present two observability criteria for BN, the first one using the semi-tensor product, a generalised matrix product, to give an algebraic condition for observability and the second one using the state transition diagram to give a structural condition on the graph in order to guarantee observability of the BN.

Presentation of Boolean Network

A Boolean network is composed of a set of Boolean variables whose state is described by a Boolean function (potentially different for each variable). This function uses the state of several other Boolean variables to compute the next state of the variable it represents. This interconnection between the variables realised by the Boolean functions can be represented as a graph (see Figure 3.2a) where the nodes represent the Boolean variables and the edges represent a dependency by the transition function. The graph is complemented by a system of state equations (see Figure 3.2b) that describes the algebraic expression of the Boolean functions.



(a) Boolean Network

$$\begin{cases} A_{t+1} &= \overline{A_t} \\ B_{t+1} &= B_t.C_t \\ C_{t+1} &= \overline{A_t} \oplus B_t \end{cases}$$

(b) System of equation

Figure 3.2. Simple example of a three-variable Boolean network described by the graph and the equation system.

Because of the non-linearity (from the point of view of linear algebra) of the logical operators of the Boolean algebra (only the operators **XOR** and **NOT** can be considered as linear), it is difficult to study Boolean networks in an algebraic way. Many study methods, especially those of the basins of attraction and cycles, use instead the state transition diagram (see Figure 3.3), which describes the behaviour of the BN on the configuration graph.

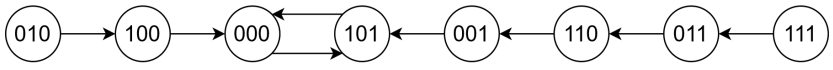


Figure 3.3. State transition diagram of the Boolean network in Figure 3.2. The Boolean variables are arranged in the order ABC for the representation of configurations.

From the definition of BN, we quickly notice that Boolean CA are a particular case of BN (Gershenson, 2002). Indeed, the lattice and the neighbourhood represent the interactions between the cells of the CA. It is a particular case of Boolean network

where the network must respect the topology of the lattice and the neighbourhood. The restrictive requirement of the Boolean function (*i.e.* two states) can be overcome by using a n-ary logical function (Cheng and Qi, 2005). Therefore CA, even non-boolean CA, can be considered as Boolean networks. However, the state transition diagram will have more nodes, e.g. k^N for a logic with k states and N variables.

Semi-Tensor Product and Observability

Described by Cheng in the early 2000s, the semi-tensor product (STP) is a generalisation of the usual matrix product when the two matrices do not have compatible sizes. Unlike the Kronecker product (tensor product) and Hadamard product (element-wise product), the STP is equivalent to the usual matrix product when the sizes of the two matrices match. In addition to its purely algebraic definition, the STP has very concrete applications such as control theory, to find a singular feedback linearisation, or to find a matrix representation of logical operators (what we will be doing thereafter) (Cheng, 2005).

In this section, we will present in detail the STP and how it is used to represent logical functions and more specifically BN. Then, we will present how Cheng and Qi use the state representation provided by the STP to verify the observability of BN. Finally, we present the drawbacks of using such a technique.

The semi-tensor product allows the product of two matrices $A \in M_{p\times q}$ and $B \in M_{m\times n}$ even when they do not have adequate sizes (*i.e.* $q = m$) but requires that the sizes of the matrices are multiples of each other (*i.e.* $\exists k \in \mathbb{N}, q = km$ or $m = kq$). If this constraint is respected then the semi-tensor product is defined by definition 3.2 (an example of the detailed calculation is presented in example 3.4) and satisfies the associativity and distribution rule with respect to matrix addition.

Definition 3.2 (Semi-tensor product (Cheng, 2005))**.** Let $A \in M_{p\times q}$ and $B \in M_{m\times n}$. The semi-tensor product is defined by:

$$A \ltimes B = A(B \otimes I_k) \in M_{p\times kn}, \text{ if } q = km$$
$$A \ltimes B = (A \otimes I_k)B \in M_{kp\times n}, \text{ if } kq = m$$

where $\otimes$ is the Kronecker product and I_k is the identity matrix in $M_{k\times k}$.

In order to be able to use the STP for BN and CA, it is necessary to be able to model the logical expressions in a matrix form. Cheng and Qi (2005) starts by defining a logical domain D_k with k states by:

$$D_k = [\![0; k-1]\!] \tag{3.15}$$

To use matrix expression each element $i \in D_k$ is identified with a vector $v_i \in M_{1\times k}$ as:

$$i \in D_k \iff v_i = \delta_i^k, \text{ with } i \in [\![0; k-1]\!]$$

where δ_i^k is the i-th column of identity matrix I_k. Therefore, in a Boolean logic (*i.e.* $k = 2$) 0 corresponds to $v_0 = \begin{bmatrix} 1 & 0 \end{bmatrix}'$ and 1 to $v_1 \begin{bmatrix} 0 & 1 \end{bmatrix}'$. In 5-state logic, state 0 corresponds to the vector $v_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix}'$ and state 3 to $v_3 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix}'$.

Remark 3.2 (Difference from Cheng and Qi definition)**.** In order to be consistent with the definitions of cellular automata given in chapter 2, we have adapted the definition of the D_k domain. In their definition, Cheng and Qi uses the fractional values $i/(k-1)$ instead of integer values, for the domain D_k, in order to relate to fuzzy logic. Furthermore, the elements of D_k were arranged in descending order. We reversed the order to correspond as well as possible to the configurations of the CA where 0 is the first value and $k-1$ the last one (with respect to the usual order).

A basic s-ary logical operator (an operator with s operands) is a mapping $\sigma\colon D_k \times D_k \times \cdots \times D_k \longrightarrow D_k$ which can be represented by a $k \times k^s$ logical matrix[2] M_σ. The application of this operator to logical variables $x^{(i)}$ is done with the STP as follows:

$$\sigma(x^{(0)}, x^{(1)}, \ldots, x^{(s-1)}) = M_\sigma \ltimes x^{(0)} \ltimes \cdots \ltimes x^{(s-1)} \tag{3.16}$$

In addition to the basic logical operators, Cheng shows that any logical function l can be modelled by a logical matrix M_l of size $k \times k^s$, where s is the number of inputs to the function. This can be determined analytically with the help of the STP, the logic matrices of the basic operators as well as various tools such as reduction matrices M_r (matrix aiming at reducing the order of a variable, for instance such as $M_r \ltimes A = A \ltimes A$) or swap matrices W (matrix which makes it possible to invert the variables, such as e.g. $W \ltimes B \ltimes A = A \ltimes B$). We will not describe explicitly the procedure for the construction of this logical matrix, but the details of the operations to construct the matrix L can be found in appendix B or in (Cheng, 2005; Cheng and Qi, 2010).

In (Cheng and Qi, 2009), the authors present a method to write a BN with n k-logical variables $x^{(i)}$ in the form:

$$x_{t+1} = L \ltimes x_t \tag{3.17}$$

where L is a $k^n \times k^n$ logical matrix and $x_t = x_t^0 \ltimes x_t^1 \ltimes \cdots \ltimes x_t^{n-1}$.

This form provides a state representation for BN (and by extension CA) even when the logical functions used are non-linear. The construction of L from logical

[2] A logical matrix is matrix representation of a binary relation. It consists in a matrix of 0 and 1 that represents the pair in a finite set. More information on section 3.3.2 and https://en.wikipedia.org/wiki/Logical_matrix

expressions is performed in a similar way to M_l. Cheng and Qi also propose a method for finding the logical functions l from the logical matrix L.

The state variable x_t represents a logical vector that describes the state of the logic variables (see example 3.4). This vector represents the node of the state transition diagram of the system at time t. Thus, L plays the same role as an adjacency matrix and we can write $L = A'$ where A is the adjacency matrix of the state transition diagram.

Example 3.4

Considering the Boolean network in Figure 3.3, we can represent its state X_t when $A_t = 0$, $B_t = 1$ and $C_t = 1$ by :

$$\begin{aligned} X_t &= A_t \ltimes B_t \ltimes C_t \\ &= \begin{bmatrix}1\\0\end{bmatrix} \ltimes \begin{bmatrix}0\\1\end{bmatrix} \ltimes \begin{bmatrix}0\\1\end{bmatrix} = \left(\begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \cdot \begin{bmatrix}0\\1\end{bmatrix} \ltimes \begin{bmatrix}0\\1\end{bmatrix} \\ &= \begin{bmatrix}1 & 0\\0 & 1\\0 & 0\\0 & 0\end{bmatrix} \cdot \begin{bmatrix}0\\1\end{bmatrix} \ltimes \begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}0\\1\\0\\0\end{bmatrix} \ltimes \begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\\0\end{bmatrix} \end{aligned}$$

From this representation, it quickly follows that the output operator of the sensor network $\mathcal{H}$ can be represented as a logical matrix as well. The logical matrix H is of size $k^q \times k^n$, where q is the number of sensors. Each column of the H represents the output associated with the configuration represented by the column (*i.e.* the number of the configuration). For example, if we observe the variable A in the BN of Figure 3.2, then y_t is $\begin{bmatrix}0 & 1\end{bmatrix}'$ when the configurations are 100, 101, 110 or 111 and $\begin{bmatrix}1 & 0\end{bmatrix}'$ otherwise. Thus the matrix H can be written as follows:

$$H = \begin{bmatrix}1 & 1 & 1 & 1 & 0 & 0 & 0 & 0\\0 & 0 & 0 & 0 & 1 & 1 & 1 & 1\end{bmatrix}$$

The state representation of the BN can be written as:

$$\begin{cases} x_{t+1} &= L \ltimes x_t = L^{t+1} \ltimes x_0 \\ y_t &= H \ltimes x_t = H \ltimes L^t \ltimes x_0 \end{cases} \tag{3.18}$$

The equation (3.19) defines the observability matrix O from L and H. Each column of this matrix represents the output sequence associated with the configuration of the column. Thus, the observability criterion of theorem 3.5 follows which ensures that each output sequence (column) is unique. However, O is not, unlike L and H, a logical matrix. The logical matrix corresponding to the output sequence O_T is obtained by the product $Y_T = y_0 \ltimes \cdots \ltimes y_{T-1}$ and by factoring by x_0. This operation is used to obtain the logic matrix associated to the output sequence but it is not necessary to calculate the observability.

$$O_T = \begin{bmatrix} H \\ H \ltimes L \\ \vdots \\ H \ltimes L^{T-1} \end{bmatrix} \tag{3.19}$$

Theorem 3.5 (Observability of Boolean Network (Cheng and Qi, 2009))**.** *The system (3.18) is observable at time T if and only if the observability matrix O_T has all distinct columns.*

The adaptability property of definition 2.7 also applies to BN. It is precisely for the observability of BN that Laschov *et al.* (2013) have given this criterion. In a logical matrix, the value of an application is given by its column, therefore the adaptability criterion is defined by the following proposition.

Proposition 3.1. *Let L be the logical matrix of a BN with N variables and k states. An output operator H is adapted for the observation of the BN if and only if:*

$$\forall i, j \in k^N \text{ with } i \neq j, Col_i(L) = Col_j(L) \implies Col_i(H) \neq Col_j(H)$$

Reconstructibility is slightly more difficult to verify than observability. Indeed, it is necessary to make sure that for the configurations which have identical columns in O_T also have an identical current state s_T. In this case, the columns in F^T must also be identical. We can therefore formulate the following theorem:

Theorem 3.6. *The system (3.18) is reconstructible at time T if and only if the matrices O_T and F^T satisfy the following property::*

$$\forall i, j \in k^N \text{ with } i \neq j, Col_i(O_T) = Col_j(O_T) \implies Col_i(F_T) = Col_j(F_T)$$

To summarise, this method provides a state representation for BN and CA. However, the size of the matrices depends on the number of configurations and not on the number of cells of the CA, as it was the case for the previous section. If we consider

N the number of cells, Q the number of sensors, k the number of states in the state space and T the observation horizon, the matrices L, H and O are respectively of size $k^N \times k^N$, $k^Q \times k^N$ and $k^Q \cdot T \times k^N$. Therefore, this method can hardly be applied if the lattice of the CA exceeds several dozen or hundreds of cells.

Graph Condition for Observability

Laschov *et al.* (2013) rewrote the results of Cheng and Qi (2009) on observability and formulated an equivalent criterion using a structural condition on the state transition diagram to prove that verifying the observability of a BN is an NP-Hard problem. This means that there is no general method to check observability with a polynomial complexity depending on the number of Boolean variables (or cells in the case of CA), except for simplifications (like the case of ACA). In this section, we will present the Laschov observability criterion and relate it with the algebraic method presented previously.

Laschov *et al.* merged together the state transition diagram and the output operator into an observability graph (see Figure 3.4). The graph associated with the state transition diagram is modified to include the output. The edges of the graph are coloured (have a certain value called colour) which represents the y_t output of the system. The output sequence is then defined as the set of colours obtained during a graph run from the initial node. For example in Figure 3.4a, if the variable A is the output and the initial node is 011 then the coloured output sequence is (*continuous*, *dashed*, *continuous*, etc.) which corresponds to the Boolean output sequence $(0, 1, 0, \ldots)$.

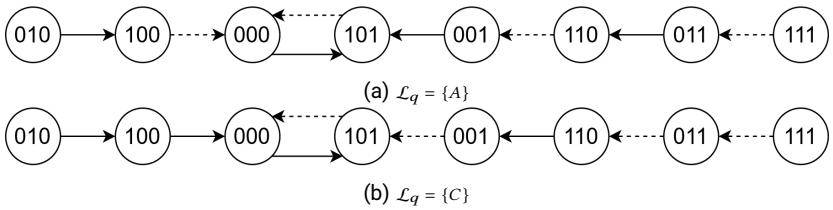


Figure 3.4. Example of an observability graph for the BN of Figure 3.2. The continuous edge represents an output of 0 and the dashed edge an output of 1. The outputs of **(a)** and **(b)** correspond respectively to the Boolean variables A and C.

The observability criterion based on this observability graph G is defined by theorem 3.7. However the notion of cycle in this theorem differs from the notion of cycle presented in section 2.2.4. The term cycle refers here to a sequence which is looped for example the cycle $2 \longrightarrow 3 \longrightarrow 4 \longrightarrow 1 \longrightarrow 2$ is considered different from the cycle $4 \longrightarrow 1 \longrightarrow 2 \longrightarrow 3 \longrightarrow 4$. If nodes 1 and 3 are associated with colour c_1 and 2 and 4 with colour c_2 then the two previous cycles have the same output sequence

$(c_2, c_1, c_2, c_1, c_2)$. It is, however, a single cycle in the sense of the definition made in section 2.2.4.

Theorem 3.7 (Observability of Boolean Network (Laschov *et al.*, 2013)). *An observability graph G is observable for some time $T > 0$ if and only if G does not satisfy any of the two following properties:*

1. *There exists a node with two input edges of the same colour.*
2. *There exist two cycles generating the same output sequence.*

For example, we can quickly see that the observability graph in Figure 3.4a is not observable because the 000 node has two input edges that have the same colour (dashed). The second graph is, however, observable as there is no node that has the same input colour twice and the only two cycles ($000 \longrightarrow 101 \longrightarrow 000$ and $101 \longrightarrow 000 \longrightarrow 101$) have different output sequence.

Unlike other observability theorems, theorem 3.7 does not give the time horizon T from which the BN is observable, it only guarantees that it exists. In the proof of this theorem, Laschov *et al* use $T_{max} = V^2 + 1$ (V the number of vertices or nodes) as the upper bound of the time horizon. This time is however very large compared to the minimal observability time. For example, the observation graph in Figure 3.4b has a maximum time horizon of $T_{max} = 65$ while it is observable for $T = 5$ (the output sequences of 111 and 011 are only distinguishable from $T = 5$).

The notion of reconstructibility is not addressed by Laschov *et al* but it follows from theorem 3.7 on observability. Indeed, **property 1**, *"There exists a node with two input edges of the same colour"*, is not necessary for reconstructibility. This property is the equivalent of the adaptability property presented in definition 2.7. If a node has two inputs of the same colour, then it is not possible, from the point of view of the output sequence, to differentiate between these two nodes but they both have the same current node. Reconstructibility is therefore not affected. Property 2 is still necessary as two different cycles necessarily have a different current node (even if they are on the same attractor cycle from section 2.2.4).

Laschov *et al.* also proposed algorithms to check the two properties necessary for observability. Property 1 is evaluated with the logic matrices L and H as in proposition 3.1. For the second property, they rely on the Jungers and Blondel (2011) algorithm which allows to check if a graph has the property 2. This algorithm constructs another graph G^2 whose nodes are pairs of distinct nodes of G and two nodes are linked by the same colour. If G^2 admits a cycle, then property 2 holds true for G. The search for the cycle is performed by a breadth-first algorithm. For an observability graph with p vertices and q possible edge colours, the complexity of this algorithm is $O(p^4 q)$. The algorithmic complexity for a BN with N variables, k states per variable and Q measured variables is $O(k^{4N+Q})$.

The previous algorithm has exponential complexity (hence not polynomial) but there could be algorithms in polynomial time. Therefore, in order to prove that observability evaluation is an NP-Hard problem, Laschov *et al.* created a special BN whose observability graph is an N-cycle with 2 colours (*continuous*, *dashed*) but only one node has the *continuous* colour. Thus, the observability problem for this BN can be reduced to a 3-variable Boolean satisfiability problem (or 3-SAT) which is a well-known NP-complete problem. Except in the eventuality where $P = NP$ (one of the millennium prize problems) and the observability problem can be reduced to an NP problem, there is no general solution in polynomial time. However, the NP-hardness does not prevent algorithms from being found in polynomial time under certain hypotheses (for example, when we consider only ACA).

In conclusion, this method allows the observability and reconstructibility of BN to be evaluated from the observability graph which is constructed from the state transition diagram and the sensor network. Like the algebraic method presented by Cheng and Qi the complexity of evaluating observability and reconstructibility is exponential as a function of the number of cells, but this method makes it possible to know whether a BN is observable without having to search, by trial and error, for a valid time horizon. The downside is that this method does not provide the observation horizon, which makes it difficult to apply this method to the monitoring of complex systems.

3.3.2 Binary Relation Representation of Cellular Automata

As we have seen in the previous section, CA can be seen as a special case of BN. Moreover, the observability and reconstructibility of BN have been widely studied for over two decades, either in an algebraic way with the use of logic matrices and the semi-tensor product (Cheng and Qi, 2009), or in a structural way with the use of the state transition diagram Laschov *et al.* (2013). These two representations are related: the logic matrix L is merely the transpose of the adjacency matrix of the state transition diagram. The criteria which result from these different representations make it possible to check the observability of the Boolean network in various ways.

With these two representations, verifying the observability is algorithmically complex: Laschov *et al.* have shown that it is an NP-hard problem. Moreover, the algorithms which verifies the observability often depend on the number of configurations (it is the case for the theorem 3.5 where each column represents a configuration) which grows exponentially according to the number of cells. This exponential complexity prevents the use of the algorithm when this number of cells exceeds several dozens or hundreds.

In order to reduce both the algorithmic and memory complexity, we can represent

the logical matrix by sparse matrices (Buluç *et al.*, 2009), an advantageous formulation for matrices that have few non-zero elements. Moreover, by using the Yale format (or Compressed Row Format) observability can be evaluated by checking that a 1D array does not contain duplicates. Therefore, the algorithmic complexity to evaluate the observability of a CA with S^N configurations is $O(S^N)$. The structural analysis of the graph, requiring to verify observability with theorem 3.7, can also have simplifications that reduce the complexity, or at least the computation time.

Even using sparse matrices, the calculation of observability or reconstructibility for CA becomes impossible when it exceeds a hundred cells. The objective is therefore to benefit from some properties of CA in order to simplify the computations of observability and reconstructibility. The first property concerns in particular CA with constraints on the initial configurations. Many initial configurations can be excluded from the observability computation because they are impossible in the concerned application case. If we take the example of section 5.2, as we know the topology of the forest and that we consider that the fire can only start on one cell of the CA, the number of initial configurations to study goes from 3^{5054} to only 5055. The second property is based on the ease of distribution of the computation of CA (Millán *et al.*, 2017). Indeed, when the number of cells is large, it could be more efficient to compute several times the local function f than the global function F once. Therefore the computation of the observability matrix O is no longer performed from the product of L and H but from a large number of simulations of the CA. The output sequence of each initial configuration is evaluated independently and the measurements are from simulations of the CA (often performed on GPU).

In order to benefit from those properties, we propose a definition of observability and reconstructibility based on binary relations. It associates two elements, here the initial configuration s_0 (or current configuration s_T if we study the reconstructibility) and the associated output sequence $\Theta_T(s_0)$. When we build this binary relation, we add the pair $(s_0, \Theta_T(s_0))$ only if the initial configuration s_0 is part of the initial configurations to study. In this formulation, the calculation of Θ_T can be done by simulation or by matrix product using L and H according to what is more efficient. Therefore, we can define the **observability binary relation** $\mathcal{R}_\Theta$ and the **reconstructibility binary relation** $\tilde{\mathcal{R}}_\Theta$ by:

Definition 3.3. The observability and reconstructibility binary relation $\mathcal{R}_\Theta$ and $\tilde{\mathcal{R}}_\Theta$ associated to the output sequence Θ_T is defined as follows:

$$\mathcal{R}_\Theta = \{(s_0, \Theta_{0,T}) | s_0 \in \mathcal{W} \text{ and } \Theta_{0,T} = \Theta_T(s_0)\} \tag{3.20}$$

$$\tilde{\mathcal{R}}_\Theta = \{(s_T, \Theta_{0,T}) | s_0 \in \mathcal{W} \text{ and } \Theta_{0,T} = \Theta_T(s_0) \text{ and } s_T = F^T(s_0)\} \tag{3.21}$$

where $\mathcal{W} \subset \mathcal{S}^{\mathcal{L}}$ represents the set of admissible initial configurations.

Remark 3.3. If we consider the whole set of initial configurations (*i.e.* $\mathcal{W} = \mathcal{S}^{\mathcal{L}}$) then $\mathcal{R}_\Theta$, M_O, Θ_T represent the same relation except that $\mathcal{R}_\Theta$ is in set representation and M_O in binary matrix form.

Theorem 3.8. *A CA $\mathcal{A}$ is **observable** (resp. **reconstructible**) by a sensor network $\mathcal{H}$ at time T if and only if the binary relation $\mathcal{R}_\Theta$ (resp. $\tilde{\mathcal{R}}_\Theta$) is injective.*

Proof. For the observability condition, the theorem follows directly from definition 2.5 which represents the writing of the injectivity of Θ, since $\mathcal{R}_\Theta$ is the relational writing of Θ.

The reconstructibility condition (2.27) in definition 2.6 may be written, using the binary relation representation, as:

$$\begin{aligned}
&\forall s'_0, s''_0 \in \mathcal{W}, \Theta_T(s'_0) = \Theta_T(s''_0) \implies s'_T = s''_T \\
&\iff \forall s'_T, s''_T \in F^T(\mathcal{W}), \tilde{\mathcal{R}}_\Theta(s'_T) = \tilde{\mathcal{R}}_\Theta(s''_T) \implies s'_T = s''_T \\
&\iff \tilde{\mathcal{R}}_\Theta \text{ is injective}
\end{aligned}$$

□

From the BN presented in Figure 3.2, we are able to construct the observability and reconstructibility relations. We will use the output operators presented in Figure 3.4, namely $\mathcal{L}_q = \{A\}$ and $\mathcal{L}_q = \{C\}$ with a time horizon $T = 5$. Table 3.1 presents all the elements of the two relations $\mathcal{R}_\Theta$ and $\tilde{\mathcal{R}}_\Theta$. As in the previous section, we notice that measuring the state of C ensures observability contrarily to the measurement of the state of A. Moreover in this example, we can also guarantee that measuring the state of A ensures reconstructibility because identical output sequences are associated with the same current configurations.

$\mathcal{L}_q = \{A\}$	$\mathcal{L}_q = \{C\}$
(000, 01010)	(000, 01010)
(001, 01010)	(001, 11010)
(010, 01010)	(010, 00010)
(011, 01010)	(011, 10110)
(100, 10101)	(100, 00101)
(101, 10101)	(101, 01010)
(110, 10101)	(110, 01101)
(111, 10101)	(111, 11011)

(a) Observability Relation $\mathcal{R}_\Theta$

$\mathcal{L}_q = \{A\}$	$\mathcal{L}_q = \{C\}$
(101, 01010)	(101, 01010)
(101, 01010)	(101, 11010)
(101, 01010)	(101, 00010)
(101, 01010)	(101, 10110)
(000, 10101)	(000, 00101)
(000, 10101)	(000, 01010)
(000, 10101)	(000, 01101)
(000, 10101)	(000, 11011)

(b) Reconstructibility Relation $\tilde{\mathcal{R}}_\Theta$

Table 3.1. Observability and reconstructibility relations for the BN of Figure 3.2 with a time horizon $T = 5$.

In order to numerically verify the observability or reconstructibility of the CA, we must first construct the relations $\mathcal{R}_\Theta$ and $\tilde{\mathcal{R}}_\Theta$ and then verify their injectivity. Therefore we propose algorithm 1 on the facing page that constructs simultaneously the relations $\mathcal{R}_\Theta$ and $\tilde{\mathcal{R}}_\Theta$ and also to check their injectivity during their construction.

When the algorithm detects that the observability or the reconstructibility is not verified, it has an exit condition without having to study all the initial configurations.

Similarly to the observability and reconstructibility relations, we could construct the adaptability relation $\mathcal{R}_A$ which associates the state $s_1 = F(s_0)$ with the output $\theta_0 = H(s_0)$. In order to guarantee adaptability, it is necessary that the relation $\mathcal{R}_A$ can be constructed without duplicates. Indeed, if there is a duplicate, then s_0' and s_0'' exist so that $s_0' \neq s_0''$ and $s_1 = F(s_0') = F(s_0'')$ and $\theta_0 = H(s_0') = H(s_0'')$. However, as with algorithm 1, we evaluate observability and reconstructibility simultaneously, the adaptability property is not necessary. Indeed, if for a given configuration s_0 observability is not verified but reconstructibility remains possible, then adaptability is not true (corollary of property (b) of proposition 2.1).

Algorithm 1: Verification of observability and reconstructibility of $(\mathcal{A}, H)$

Function AssessObsRec $(\mathcal{W}, F, \Theta_T)$
- **Inputs** : A set of admissible initial configuration $\mathcal{W}$;
  a global transition function F;
  an output sequence operator Θ_T
- **Outputs** : The observability Hashmap **Obs**;
  the reconstructibility Hashmap **Rec**;
  the observability status **is_Obs**;
  the reconstructibility status **is_Rec**
- Declare ***Obs*** and ***Rec*** as two empty Hashmap
- Declare ***is_Obs*** and ***is_Rec*** as two boolean variables initialised to **true**
- **foreach** $s_0 \in \mathcal{W}$ **do**
  - $(\Theta_{0,T}, s_{T-1}) \leftarrow \Theta_T(s_0)$
  - $s_T \leftarrow F(s_{T-1})$
  - **if** ***Obs***$[\Theta_{0,T}]$ *already exists* **then**
    - **is_Obs** $\leftarrow$ **false**
  - **else**
    - **Obs**$[\Theta_{0,T}] \leftarrow s_0$
  - **if** ***Rec***$[\Theta_{0,T}]$ *already exists* **then**
    - **if** ***Rec***$[\Theta_{0,T}] \neq s_T$ **then**
      - **is_Rec** $\leftarrow$ **false**
  - **else**
    - **Rec**$[\Theta_{0,T}] \leftarrow s_T$
- **return** ***Obs***, ***Rec***, ***is_Obs***, ***is_Rec***

In order to construct the observability relation, the output sequence is computed

for each of the possible initial configurations $s_0 \in \mathcal{W}$. However, to obtain the output sequence $\Theta_T(s_0)$ we need the configurations from s_0 to s_{T-1}, so we can take advantage of s_{T-1} to compute s_T in order to build the reconstructibility relation. As for the evaluation of the injectivity, we use a specific data structure called *Hashmap* to record the relations $\mathcal{R}_\Theta$ and $\tilde{\mathcal{R}}_\Theta$. This type of data structure stores a value with a key and to guarantee the uniqueness of the key. In this manner, by using the output sequence as a key and the initial configuration (or current configuration) as a value we are able to verify two things. The first is to make sure that each output sequence is associated to only one configuration and thus to check the injectivity of the observability and reconstructibility relations. The second is to be able to easily find the configuration related to the output sequence and therefore know the initial configuration when measurements are performed on the system.

Algorithm 1 is separated into two parts, the first where the output sequence and the current configuration are computed and the second where these values are stored in the Hashmap. The first part has a complexity of $O(TF)$ where T is the time horizon and F is the complexity of the global transition function F. If we consider that the global transition function F is evaluated as the local function applied to all cells, then its complexity is of the order of $O(N)$. However, due to the ease of parallelisation of these operations, especially on a GPU, then the computation time is largely reduced because a very large number of these operations (or even all of them) are evaluated at the same time when the computation is performed on a GPU. Moreover, the observability horizon T is at worst of the order of N in an observation problem. Indeed, if this horizon is much larger than N, then the disturbance we are trying to estimate (a forest fire, a road traffic, etc.) has had time to propagate over the whole system before we are able to reconstruct it. This is even more true for dimensions greater than 1 where the dynamics propagates faster compared to the number of cells. The second part of the algorithm has a constant complexity of $O(1)$ as the operations of read and writes of Hashmap have both a complexity of $O(1)$. Finally, the total algorithmic complexity is $O(WN^2)$ where W is the number of admissible configurations.

If we consider that all initial configurations are studied (*i.e.* $\mathcal{W} = \mathcal{S}^{\mathcal{L}}$) then the complexity of the algorithm is $\mathcal{O}(k^N N^2)$. Compared to the algorithm that verifies observability for affine CA presented in section 3.2.3, this algorithm has a greater complexity. The algorithmic complexity for verifying observability is polynomial for affine CA $\mathcal{O}(TQN^{\omega-1})$ and exponential for non-linear CA $\mathcal{O}(k^N N^2)$, with N representing the number of cells, Q the number of sensors, k the number of states and T the time horizon.

3.3.3 State Estimator for Non-Linear Cellular Automata

In this section we will define a state estimator for non-linear CA in order to estimate the state of the CA from the measurements. It will be notably necessary for the decentralisation of the observation that we will present in the next section. The state estimator that we will present is similar to those that we can find in (Yang *et al.*, 2020) and (Braga-Neto, 2011). It is defined for Boolean networks and provided an estimation of the current state of the system. We will see with theorem 3.9 that reconstructibility is necessary to correctly estimate the state of the system. We will first define the tools needed to build the state estimator and prove theorem 3.9.

Tools for the State Estimator

The state vector x_t defined in section 3.3.1 for the observation of BN represents the configuration of the system at time t. This state vector has a single 1 coordinate which represents the current configuration, while all other coordinates are 0. It may be extended to a vector with more non-zero coordinates to represent a set of configurations, and to coordinates which are greater than 1 to put different weights on some of these configurations. This extension is for instance widely used in graph theory with the adjacency matrix which is generalised to represent the number of paths between two nodes. Such an extended vector x_t represents therefore an equivalent set of configurations which is explicitly given in the following definition.

Definition 3.4. The state vector x_t can be written as a subset X_t of $\mathcal{S}^{\mathcal{L}}$ by:

$$X_t = \{s_t \in \mathcal{S}^{\mathcal{L}} | s_{t|k} = i \wedge x_t^{(i)} = 1\} \tag{3.22}$$

where $s_{t|k}$ is the decimal value of the configuration state (see section 2.2.4 on *State Transition Diagram and Attractor*) and $x_t^{(i)}$ is the i-th element of the vector x_t.

From this set, we can also compute the image sets of the functions F, H and Θ (simply by computing the function on each of the elements of X_t). For instance

$$X_{t+1} = F(X_t) = \{F(s_t) | s_t \in X_t\}$$

The set $X_{t+1} = F(X_t)$ is equivalent to the state vector $x_{t+1} = Lx_t$. If F is not reversible, then there exists $s_0', s_0'' \in X_0$ and $s_0' \neq s_0''$ such that $F(s_0') = F(s_0'')$. In that specific case, the element corresponding to $s_1 = F(s_0') = F(s_0'')$ in x_1 will have the value 2, one for $F(s_0')$ and one for $F(s_0'')$. We could normalise the result in order to have values in $\{0, 1\}$. However, in the case of the state estimation problem and algorithm, it will be useful to have these extended values (greater than one) to construct a good estimate of the cell state as it provides more information on the number of

initial configurations that have converged toward s_1. In terms of set definitions, this means that X_1 contains the configuration s_1 twice, once for $F(s'_0)$ and once for $F(s''_0)$. This is a particular definition of a set with several iterations of the same element. We can adapt definition 3.4 to allow its use with non-unitary coefficients by using the following notation:

$$X_t = \{s_t \in \mathcal{S}^{\mathcal{L}} | s_{t|k} = i \wedge x_t^{(i)} \geq 1\}$$

For the construction of the state estimator, we will need an operation, similar to the intersection of the sets X_t, for the state vectors x_t. For this, we will use the Hadamard product $\odot$ (or element-wise product) which multiplies element by element the two operands (the two operands must have the same size). Therefore we propose the following to compute the intersection between the two set from the state vector:

Proposition 3.2. *Let x'_t and x''_t be two state vectors and X'_t, X''_t their set representations. The intersection X_t between these two sets can be computed from the state vector with the help of the Hadamard product as:*

$$x_t = x'_t \odot x''_t$$

Proof. Let X_t be the set representation of $x_t = x'_t \odot x''_t$ and x'_t. Equation (3.22) in 3.4 may be written with the set representations:

$$\begin{aligned} X_t &= \{s_t \in \mathcal{S}^{\mathcal{L}} | s_{t|k} = i \wedge x_t^{(i)'} \cdot x_t^{(i)''} = 1\} \\ &= \{s_t \in \mathcal{S}^{\mathcal{L}} | s_{t|k} = i \wedge x_t^{(i)'} = 1 \wedge x_t^{(i)''} = 1\} \\ &= \{s_t \in \mathcal{S}^{\mathcal{L}} | s_t \in X'_t \wedge s_t \in X''_t\} \\ &= X'_t \cap X''_t \end{aligned}$$

□

In the case where the vectors have non-unitary coefficients then the Hadamard product will multiply these coefficients. If a configuration appears twice in a set and twice in another set, the intersection with the Hadamard product will produce a set where this configuration appears 4 times. This is why we impose that at least one of the vectors x_t use in the Hadamard product has unitary values. It will avoid increasing the coefficient in the intersection vector. In the next section, when we will study the state estimator, the reciprocal of the output operator (H') is a state vector with unitary value, so we will be able to use the Hadamard product as an intersection operator.

Definition of the State Estimator

The state estimator is constructed from the measurements made on the system. The output y_t which represents the measurement of the system is considered as an input to

the state estimator. The latter represents the set of configurations that are consistent with the measurements made on the system. Thus, the set of estimated configurations $\hat{X}_t$ evolves at each acquisition of a new measurement: all the configurations which are not compatible with the measurement are withdrawn from $\hat{X}_t$. Considering that at time $t = 0$ the set of possible configurations is the set of admissible configurations $\hat{X}_0 = \mathcal{W}$, we can therefore define the evolution of the state estimator as :

$$\hat{X}_{t+1} = \{F(s_t) | s_t \in \hat{X}_t \wedge H(s_t) = y_t\} \tag{3.23}$$

Using the reciprocal function of the output operator H, we can determine the set of configurations s_t that have the output y_t. In the case of a logical matrix, the reciprocal is the transpose of H. Hence, $\tilde{x}_t = H' y_t$ is the state vector that has, as its element, all configurations which have y_t as their output. We simply adapt (3.23) in order to use $\tilde{X}_t$ the set representation of $\tilde{x}_t = H' y_t$:

$$\hat{X}_{t+1} = \{F(s_t) | s_t \in \hat{X}_t \cap \tilde{X}_t\} \tag{3.24}$$

From (3.24) we propose the definition 3.5 which adapts the state estimator in the form of a state representation.

Definition 3.5. Let $\mathcal{A}$ be a CA defined by the logical matrix L and observed by a sensor network $\mathcal{H}$ defined by the logical matrix H. The estimation $\hat{x}_t$ of the state x_t is realised through the measurements y_t by:

$$\begin{cases} \hat{x}_{t+1} & = L(\hat{x}_t \odot H' y_t) \\ \hat{x}_0 & = \mathbf{1}_{k^N} \end{cases} \tag{3.25}$$

where $\mathbf{1}_{k^N}$ is a column vector of size k^N compose of 1.

The estimated state vector $\hat{x}_t$ is a set of the estimated configuration of the system. From this we can estimate the state of the cell c_i by using the output operator $H^{(i)}$ which measures the cell directly. The estimate of the cell c_i equals $\hat{x}_t^{(i)} = H^{(i)} \hat{x}_t$ and is a column vector where each coordinate represents the number of configurations in $\hat{X}_t$ that have the corresponding state value (state 0 for the first coordinate and so on). Example 3.5 hereafter details this for the BN of Figure 3.2.

Example 3.5

Consider the BN defined in Figure 3.2 and the estimated set $\hat{X}_t = \{001, 111, 101\}$. From $\hat{X}_t$ we can easily see that the node A has the state 1 for 2 out of the 3 configurations, the node B for 1 out of 3 configurations and the node C for every configuration. We can compute the same results from $H^{(A)}$, $H^{(B)}$, $H^{(C)}$ and $\hat{x}_t$

which are defined by:

$$\hat{x}_t = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}'$$
$$H^{(A)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$
$$H^{(B)} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$$
$$H^{(C)} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

From these elements we can calculate $\hat{x}_t^{(A)}$, $\hat{x}_t^{(B)}$ and $\hat{x}_t^{(C)}$ the estimates of cells A, B and C at time t by :

$$\hat{x}_t^{(A)} = H^{(A)}\hat{x}_t = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \implies A_t = 0 \text{ for 1 conf. and } A_t = 1 \text{ for 2 conf.}$$
$$\hat{x}_t^{(B)} = H^{(B)}\hat{x}_t = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \implies B_t = 0 \text{ for 2 conf. and } B_t = 1 \text{ for 1 conf.}$$
$$\hat{x}_t^{(C)} = H^{(C)}\hat{x}_t = \begin{bmatrix} 0 \\ 3 \end{bmatrix} \implies C_t = 0 \text{ for 0 conf. and } C_t = 1 \text{ for 3 conf.}$$

Lemma 3.1. *Consider an initial configuration, unknown to the state estimator, and the associated output sequence Θ at time $t > 0$. The state estimator $\hat{x}_t$ constructed from the output sequence corresponds to all configurations that have this output sequence. Formally we can write this as :*

$$\forall s_0, s_t \in \mathcal{S}^{\mathcal{L}} \text{ such that } s_t = F^t(s_0), \Theta_t(s_0) = \Theta \implies s_t \in \hat{X}_t$$

Proof. By definition (3.23) implies that if $\hat{X}_t$ respects lemma 3.1 then $\hat{X}_{t+1}$ also respects the lemma 3.1 because:

$$s_t \in \hat{X}_t \wedge H(s_t) = y_t \implies \forall n \in [\![0, t]\!], H(s_n) = y_n \implies \Theta(s_0)_{t+1} = \Theta_{t+1}$$

By definition, $\hat{X}_1$ respects the lemma 3.1 as $\Theta_1 = (y_0)$. Therefore, with an inductive reasoning, we conclude that the lemma 3.1 holds for all $t > 0$. □

From lemma 3.1 and definition 2.6 on reconstructibility, we know that at some point T there is only one current configuration x_t that respects the output sequence given as input to the state estimator. Thus we give the theorem 3.9 that guarantees perfect estimation of the system if the reconstructibility is insured.

Theorem 3.9. *If a CA is reconstructible at time T then the state estimator of definition 3.5 gives a perfect estimate of the current state at time T (i.e. $\hat{x}_T = x_T$).*

This state estimator has two major problems. The first one is the use of logical matrices which implies an exponential complexity with respect to the number of cells. The use of the set representation $\hat{X}_t$ permits to use the paradigm proposed in section 3.3.2 with the reduction of the plausible initial configurations. Even then, the state estimator is meant to be used during the observation phase, unlike observability and reconstructibility which are evaluated beforehand. In order to use the state estimator for monitoring complex systems, the computation time of an iteration (*i.e.* $\hat{x}_{t+1}$ from $\hat{x}_t$) must be smaller (or even much smaller) than the time period of the system evolution.

The second problem is the robustness of the state estimator to error, and particularly to measurement error. Indeed if the sensors provide, at a given time t, an erroneous measurement y_t then the state estimator will not be able to correctly estimate the current state. The inverse output operator $H'(y_t)$ includes all the states that have y_t as output, but as the measurement is incorrect, the current state of the system is not in $H'(y_t)$. This non-robustness to errors does not allow the use of this state estimator for physical system observation. To increase the robustness of the state estimator Guo *et al.* (2018) uses a stochastic output operator to model the probability of sensor error.

In this section we have given a state estimator for non-linear CA. We also proved that the state estimator converges to the state of the system when reconstructibility is satisfied. However, this state estimator has the same algorithmic and memory complexity problems as the study of observability and reconstructibility of BN itself. These complexity issues and the robustness problem do not allow the use of the approach for monitoring real large-scale physical systems.

3.4 Decentralisation of Observability and Reconstructibility

In the previous two sections, we were able to define observability and reconstructibility criteria for additive, affine and non-linear cellular automata. The Kalman criterion for ACA can easily be evaluated on a computer for CA for several thousand cells. In the case of non-linear CA, although further optimisations could be made, unless the number of plausible initial configurations is drastically reduced, it is simply impossible to evaluate observability beyond a hundred cells. In this section we will propose a

method to reduce the algorithmic complexity by dividing the CA into many smaller CA.

Considering cellular automata as Boolean networks, the links between cells being determined by the neighbourhood, CA are not very connected (few links between cells). This is particularly true for one-dimensional CA where the number of links is 2 compared to 4 or 9 for two-dimensional CA (depending on the chosen neighbourhood). Therefore it seems possible to decompose the observability problem into smaller ones with the goal to drastically reduce the algorithmic complexity. The observability would be computed for each part and be extended to the whole CA. The complexity factor would be reduced from the size of the CA to the size of its part. To achieve this, sensors and subdivisions must be judiciously chosen to ensure observability or reconstructibility of any cell in the CA.

The distribution of observability and reconstructibility has recently been studied for large BN (Zhang and Johansson, 2020). The authors grouped the highly connected nodes into sub-BN whose observability calculation is simpler. Interactions with other sub-BN are modelled as inputs as for BCN. In the case where the network composed of sub-BN is acyclic, the observability of all sub-BN implies the observability of the BN. In the cyclic case, the generalisation of the observability of the sub-BN to the observability of the initial BN is more complex. This method is very poorly applicable to CA because cells are mutually dependent on their neighbours, which directly implies a cyclic relationship between the cells of the CA.

In this section, we first propose a definition for the sub-CA used for decentralisation of observability and reconstructibility analysis. Then, we present a method adapted to the observation of one-dimensional CA (the cost for observation being too high for two-dimensional CA) as well as several ideas for improvements in the decentralisation of observability and reconstructibility.

3.4.1 *Observability and Reconstructibility of Sub-CA*

In order to reduce the algorithmic complexity, we will divide the observability problem into several observability problems of lower complexity. It consists in dividing the $\mathcal{L}$ lattice of $\mathcal{A}$ into several smaller lattice $\mathcal{L}^{(i)} \subset \mathcal{L}$. We call the CA defined on the subdomain $\mathcal{L}^{(i)}$ the sub-CA $\mathcal{A}^{(i)}$ of $\mathcal{A}$.

Sub-CA do not have any more autonomous dynamics: they are no longer isolated. Their boundary conditions represent interactions with cells of the bigger CA and will be referred to as **unknown boundary conditions**. As these cells are not within the sub-CA, they belong to other sub-CA that estimate their values. Neighbouring sub-CA will thus mutually exchange the values of the boundary cells. The boundary conditions of the sub-CA can be considered as inputs and we can define the sub-CA

as a CA with inputs.

Definition 3.6. A sub-CA $\mathcal{A}^{(i)}$ over a domain $\mathcal{L}^{(i)} \subset \mathcal{L}$ of a CA $\mathcal{A}$ is defined by (3.26) where $s_t^{(i)}$ and $\theta_t^{(i)}$ are respectively the configuration and the output of the sub-CA at time t. $b_t^{(i)}$ represents the value of the boundaries cells at time t. F and H have been adapted to include the subdomain $\mathcal{L}^{(i)}$ and the $b_t^{(i)}$ boundaries as inputs.

$$\begin{cases} s_{t+1}^{(i)} &= F\left(s_t^{(i)}, b_t^{(i)}\right) \\ \theta_t^{(i)} &= H\left(s_t^{(i)}\right) \end{cases} \tag{3.26}$$

These boundaries can a priori take any admissible state values at each time step. Therefore we have to define a new notion of observability and reconstructibility for CA with unknown boundary conditions. The observation of a system with unknown inputs has already been studied for linear systems (Basile and Marro, 1969) but the results cannot be applied to non-linear CA. We therefore find another criterion using the finiteness of the set of possible inputs. For a specific evolution of the bigger CA, the boundary conditions of the sub-CA are well known and therefore the state configuration $s_t^{(i)}$ has a fixed trajectory. This trajectory may be different for each initial configuration of the bigger CA. This is the reason why we consider it necessary to check the observability or reconstructibility for each fixed boundary trajectory. We propose the following proposition for the observability or reconstructibility of a sub-CA.

Proposition 3.3. *A sub-CA $\mathcal{A}^{(i)}$ with **unknown boundary conditions** is observable (reconstructible) by a sensor network $\mathcal{H}^{(i)}$ at time T_i if and only if the sub-CA is observable (reconstructible) for every admissible boundary trajectories.*

Proposition 3.3 guarantees observability and reconstructibility for the sub-CA $\mathcal{A}^{(i)}$ regardless of the initial configuration of the CA $\mathcal{A}$. Thus the state of all the cells of the sub-CA is known at time T_i. The corollary 3.3 follows immediately from this proposition. If the set of observable (reconstructible) sub-CA $\mathcal{A}^{(i)}$ covers the whole CA $\mathcal{A}$ (*i.e.* $\bigcup_i \mathcal{L}^{(i)} = \mathcal{L}$) then the set of cells of $\mathcal{L}$ are observable (reconstructible).

Corollary 3.3. *A CA $\mathcal{A}$ is observable (reconstructible) if there exists a set of observable (reconstructible) sub-CA $(A^{(i)})_{i\in\mathbb{N}}$ such that $\bigcup_{i\in\mathbb{N}} \mathcal{L}^{(i)} = \mathcal{L}$.*

Remark 3.4. If the sub-CA do not cover the whole CA, then only a region of the CA is observable (reconstructible), we can speak of regional observability or reconstructibility (Fekih and Jai, 2006). However, this is a special case of regional observability where the sensors are placed directly in the region to be observed and the dynamic of the

system outside the region is not modelled (except through the boundaries). In some cases, sensors placed outside the region of observation ensure the observability of this region. Note also that some of the results for the regional controllability (Dridi *et al.*, 2019, 2020) of CA could be transposed to observability by duality of the two properties.

The placement of the sensors is more important in the case of the decentralised observation than in the classical approach because the state values at the boundary must then be either measured or estimated. In the case where an estimation is provided to the sub-CA, it may happen that the observable time horizon is not reached yet and the estimation is not yet perfect. We discuss this issue in section 3.4.3 when a probabilistic sub-CA model is used to cope with this uncertainty of the estimation. In the next section, we consider first the case with measured boundary cell state values.

3.4.2 *Deterministic Method for Distribution*

To ensure the observability or reconstructibility of a sub-CA with proposition 3.3, it is necessary to measure the value of boundary cells at each time step. Therefore, for each sub-CA we must at least measure cells that are boundaries for other sub-CA. However, more cells could be measured, in order to insure the observability or reconstructibility. The value of those measured cells are both used in the observability criterion and as the boundary for the neighbouring sub-CA.

For one-dimensional CA, the sensors can be placed directly on both sides of the sub-CA and thus act as a sensor for the sub-CA and as a boundary cell sensor for the neighbouring sub-observers (see Figure 3.5). In this way, each sensor has the dual purpose of measuring the boundary cells and estimating the cells of the CA. Of course, if this sensor is not sufficient to ensure observability or reconstructibility, other sensors can be added within the sub-CA but, then, their sole purpose is to estimate the state of the sub-CA.

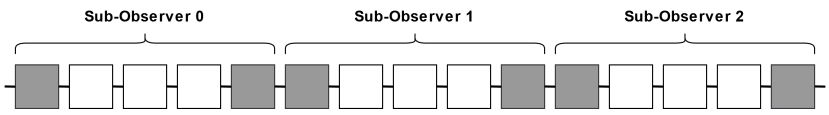


Figure 3.5. A one-dimensional CA subdivided into 3 sub-CA. Grey cells are measured by a sensor.

This method significantly reduces the complexity of the observability of one-dimensional CA because the subdivision reduces the number of cells whose observability needs to be checked. Consider a one-dimensional CA with N cells and k states. If we divide this CA in j equal parts, then the number of initial configuration goes from k^N to $k^{\frac{N}{j}}$. However, we have to evaluate the observability for each possible

boundary trajectory. For 2 boundary cells and a time horizon T there are k^{2T} possible boundary trajectories. Finally, the total number of operations to compute the observability is $k^{\frac{N}{j}} \times k^{2T} = k^{\frac{N}{j}+2T}$. If the sub-CA that compose the CA are not of the same size, then observability must be checked independently for each sub-CA which will increase the complexity linearly to the number of sizes.

This method works very well for one-dimensional CA because the size of the boundary is independent of the number of cells in the CA: there are always 2 boundary cells. However, in two dimensions (or more) the number of boundary cells increases with the number of cells. For a $N \times N$ sub-CA, there are $4N$ boundary cells for a Von Neumann neighbourhood and $4(N+1)$ boundary cells for a Moore neighbourhood. The more boundary cells there are, the more boundary trajectories have to be considered and the more operations are needed to assess the observability. For a $N \times N$ sub-CA with Von Neumann neighbourhood under a time horizon T, there is k^{N^2+4NT} operations. In addition, the minimal sensor density also increase compared to 1D CA. For a $N \times N$ sub-CA, it is necessary to measure the state of the $4(N-1)$ cells on the edge which are boundary cells for the other sub-CA. The sensor density for a sub-CA is $4(N-1)/N^2$ against $2/N$ for a one-dimensional sub-CA of size N.

3.4.3 Probabilistic Model for Estimated Boundaries

In order to use the decentralisation of observability and reconstructibility in the two-dimensional case, an alternative to the measurement of boundary cells must be found. The idea is to provide, to the sub-CA, only an estimate of the state of these cells. However, as long as the observation horizon is not reached (*i.e.* $t < T$), the state of the sub-CA is not perfectly known, *i.e.* there are still several initial configurations which have the same output sequence Θ_t. Using the state estimator given in definition 3.5 we can deduce an estimate for the state of cell on the boundary. Giving a value for the boundaries based on the statistical majority will lead to not considering the minorities, certainly less probable but still existing. In order not to lose information in the process, the proportion of each state value should be provided for each boundary cell. This formulation leads necessarily to the modelling of the sub-CA which have several potential evolution depending on the proportion of the state of the boundaries. This is precisely the case for probabilistic CA (PCA) (see section 2.2.3 on *Probabilistic Cellular Automata*) where the estimated state of the boundaries is used as a probability. This leads not only to a new model for sub-CA but also to a new probabilistic approach to observability and reconstructibility.

Probabilistic observability differs from the observability given in definition 2.5 in that for a given initial configuration there may be several output sequences $\Theta(s_0)$. In the probabilistic case, the objective is still the same: to identify the initial configuration

from an output sequence, but this is more complicated due to the non-determinism of F. In the case of probabilistic sub-CA, we have the advantage of having a deterministic output operator because the system we observe is deterministic (if we do not seek to handle measurement error as explain as the end of section 3.3.3). Fornasini and Valcher (2019) studied this special case (*i.e.* probabilistic F and deterministic H) for probabilistic Boolean networks (PBN). They provided a definition for observability and two definitions for reconstructibility, one strong and one weak. Several algorithms to check these three criteria were also given.

Definition 3.7 (Observability and Strong/Weak Reconstructibility)**.** Fornasini and Valcher (2019) define the following notion for observability and reconstructibility of PBN.

- A PBN is **observable** at time T if for every admissible output sequence Θ_T the initial configuration s_0 can be uniquely identified.
- A PBN is **strongly reconstructible** at time T if for every admissible output sequence Θ_T the current configuration s_T can be uniquely identified.
- A PBN is **weakly reconstructible** at time T if for every admissible output sequence Θ_T the intermediate configurations s_t, for $0 < t < T$, can be uniquely identified.

The observability and the strong reconstructibility are similar to those we defined in section 2.3.3. Weak reconstructibility, on the other hand, is a proper notion due to the non-determinism of F. Indeed, if we know s_t for $t < T$ then we do not necessarily know s_T because of the non-determinism of F. The approach used to model PBN is similar to (3.18). The PBN logical matrix L is defined as a discrete function which is equal to a deterministic logical matrix L_i with a probability p_i as follows:

$$L = \begin{cases} L_1 & \text{with a probability } p_1 \\ \dots & \\ L_M & \text{with a probability } p_M \end{cases} \tag{3.27}$$

As sub-CA models a deterministic system with unknown input by a probabilistic model, there are as many different L_i as there are different inputs. Thus k^B inputs for a sub-CA with B boundary cells and k states.

To verify observability and reconstructibility, Fornasini and Valcher do not check if all the output sequences of the initial configurations are different but they construct the set $\mathcal{X}(\Theta_T)$ which contains all initial configurations s_0 which have Θ_T as output sequence. $\mathcal{X}(\Theta_T)$ is constructed as the intersection of $\mathcal{X}_t(y_t)$ which represents the set of configurations which have y_t as their t-th output. To do this, they use H' to find all the configurations s_t that have y_t as output and then they use the mapping $\mathcal{P}^t$ to

find all the antecedents s_0 of the configurations s_t. Therefore $\mathcal{X}(\Theta_T)$ is constructed as:

$$\mathcal{X}(\Theta_T) = \bigcap_{t=0}^{T-1} \mathcal{X}_t(y_t) = \bigcap_{t=0}^{T-1} \mathcal{P}^t(H' y_t) \tag{3.28}$$

If $\mathcal{X}(\Theta_T)$ is either empty or a singleton then the output sequence is uniquely associated to a single configuration. This approach works well algorithmically as there are many exit conditions (*e.g.* if for a given output sequence, $\mathcal{X}$ has more than two or more elements then observability is not checked). However, the construction of the predecessors with $\mathcal{P}$ can be complex when the number of matrix L_i is large. Similarly, the number of possible output sequences is at least as large as the set of initial configurations, which necessarily results in high computational complexity for general CA.

Once observability or reconstructibility has been verified, the state of the boundaries must be estimated. The state estimator is made in a similar way as the deterministic one. The main differences are that the logical matrices are Markovian matrices and that the weights are probabilities. For the state vectors, these are the probabilities of each configuration and for the matrices they are the probabilities of transitions from one configuration to another (*i.e.* $L = \sum p_i L_i$). However, it is necessary to normalise the Hadamard product so that the result remains a probability vector. Convergence to a unique current configuration s_t has not been demonstrated in this work but we have the intuition that strong reconstructibility is a sufficient condition. Weak reconstructibility does not seem to be relevant for this state estimator but we believe it can be adapted for it.

In conclusion, we have shown that the use of a probabilistic sub-CA model provides a solution for the decentralised approach to observability and estimation of the deterministic sub-CA. However, the probabilistic CA model raises new issues such as the new definitions of observability and reconstructibility. We have not been able to go further in the study of probabilistic sub-CA, mainly due to lack of time, but we have the intuition that they could be very useful for the observation of two-dimensional non-linear CA.

3.5 Conclusion

In this third chapter, we have given different criteria to evaluate the notions of observability, reconstructibility and adaptability proposed in the previous chapter. In the first section, we focused on ACA and in the next two other sections we studied non-linear CA.

The use of ACA provides a way to assess the injectivity of the output sequence using linear algebra. The observability criterion is then a transposition of the Kalman criterion from classical control theory. The adaptability and reconstructibility criteria are verified using the kernel of the associated operators. The use of ACA provides criteria with polynomial algorithmic complexities that allow the evaluation of these criteria with a large number of cells. However, ACA represent only a very small part of all possible CA. Therefore, the case of non-linear CA is discussed in the second part of this chapter.

In this second part, we first made the link between BN and CA. This allowed us to transpose the results concerning the observability and reconstructibility of BN to CA. We then proposed a different approach that takes advantage of the regularity of the lattice to propose better observability and reconstructibility criteria. These criteria can be used on any CA derived from definition 2.1, but the exponential algorithmic complexity makes it difficult to use for CA with more than a dozen cells.

In the last part, we proposed a method for the decentralisation of observability analysis. This method split the observability and reconstructibility problems in order to reduce the algorithmic complexity. The decentralisation works well only for one-dimensional CA. For two-dimensional CA, we suggested an approach using a probabilistic CA for the state estimator. This should reduce the number of sensors in each sub-CA that are used to measure the boundary cells state values.

Concerning section 2.2.3, the main contributions of this chapter are the use of the Kalman criterion for CA and the definition of the reconstructibility and adaptability criteria for ACA. For non-linear CA, the contribution of the thesis lies in the transposition of the observability and reconstructibility criteria from BN to CA. The relational approach and the decentralisation of observability are also two major contributions of the thesis as they easily reduce the algorithmic complexity in particular cases. All these results have been published in (Plénet *et al.*, 2022b) (currently in press) for ACA and in (Plénet *et al.*, 2022a) for non-linear CA.

In chapter 4 we will study observation through the synchronisation of CA. The approach used will be different from the one used here. The synchronisation will be used as a state estimator without checking the reconstructibility, as we did in section 3.3.3. The performance of the state estimator will be studied statistically through a large number of simulations. chapter 5 will provide illustrations of observation analysis and estimation developed in this chapter, on large and physically-motivated examples: a simplistic forest fire propagation model, a road traffic model inspired by **R184** and a high quality random number generator. These examples will be used to apply the different criteria presented throughout this chapter, but also to compare them with the results of chapter 4.

Chapter 4

Synchronisation as State Estimator

Résumé

Dans ce chapitre, nous présenterons la synchronisation des AC et nous l'étudierons comme un estimateur d'état. Cette approche est sensiblement différente de celles présentées au chapitre précédent, car la bonne reconstruction de l'état par l'estimateur n'est pas caractérisée par l'observabilité ou la reconstructibilité, mais par une étude statistique.
Dans un premier temps, nous présenterons la synchronisation des automates cellulaires de Bagnoli et nous détaillerons en quoi celle-ci est très proche de la formulation d'un estimateur d'état. Ensuite, nous caractériserons la propagation de l'erreur initiale dans les AC élémentaires lorsque l'erreur initiale est faible. Finalement, nous présenterons une nouvelle méthode de synchronisation lorsque l'erreur initiale est faible et nous la comparerons à la précédente.

Contents

4.1 Introduction

In the previous chapter, we defined observability and reconstructibility criteria for additive, affine and non-linear CA. These two notions guarantee that the state of the system can be deduced from the measurements made by the sensors. In section 3.3.3 we described a state estimator which makes it possible to determine the state of the system while it is being measured. It depends on the verification of the reconstructibility criterion that is a necessary condition to ensure the good convergence of the state estimator. However, the algorithmic complexity makes it difficult to use these tools for CA with a large number of cells. In this chapter we present a different method. We first define a state estimator and then, we evaluate its observation *"quality"* statistically from a large number of simulations. For this purpose, we will use the CA synchronisation as a state estimator.

CA synchronisation is a direct application of the synchronisation of physical systems. It is a concept that has been studied for many centuries and has recently been adapted for CA. The study of the synchronisation of two systems consists of pairing two distinct physical systems by some kind of coupling and study whether their states become identical or not. The most famous example of synchronisation was discovered by Christian Huygens in 1965 when he found that two unsynchronised pendulums (*i.e.* with an arbitrary phase) placed on the same surface would eventually synchronise (in phase opposition for Huygens's discovery but more recently studies by Fradkov and Andrievsky (2007) have shown that in-phase synchronisation can exist).

When we study the synchronisation of CA, there are two identical CA with different initial configurations. The coupling between the two CA is achieved by coupling some cells from the "driver" to the "replica". This is called unidirectional coupling. When a cell is coupled, the state of this cell on the driver is copied to the same cell on the replica. The synchronisation between the two CA varies according to the nature of the coupling and the methods used. Dogaru *et al.* (2009) studies the synchronisation of chaotic ECA using a single coupling cell and characterises the synchronisation according to the number of cells and the ECA rule. Urıas *et al.* (1998) studies the synchronisation of linear ECA with several coupling cells and characterises the position of the coupling cells to ensure full synchronisation of the replica on the driver. This approach is similar to the one we presented in section 3.2 when studying ACA. On the contrary, Bagnoli *et al.* (2010) study synchronisation of one-dimensional CA (mainly ECA and totalistic CA) using a coupling probability which represents the probability of a cell of the CA to be a coupling cell. The results are not characterisations of the synchronisation but rather statistical results that identify the critical probability for synchronisation to be achieved. This critical probability can also be approximated using the CA Lyapunov exponent (Bagnoli and Rechtman,

1999).

In this chapter we will start with a general presentation of CA synchronisation by presenting the different properties of synchronisation. Then, we will present how synchronisation can be used as a state estimator and its link with the state estimators of classical control theory. Finally, we will introduce a sensor placement method to improve synchronisation performance for CA with small initial error. We will compare the two synchronisation methods for the observation of ECA.

4.2 Generalities on the Synchronisation of Cellular Automata

Throughout this chapter, we will focus on the synchronisation proposed by Bagnoli *et al.* (2010) as it combines the use of a mobile sensor network, non-linear CA and also (and most importantly) a statistical study of the synchronisation performance. This leads to a low algorithmic complexity compared to the evaluation of observation performance. In (Bagnoli *et al.*, 2010), the synchronisation of the replica with the driver is achieved through copying the state of some cells (called coupled cells) of the driver in the matching cells of the replica. The coupled cells can change at each time step, which allows the representation of mobile sensors.

4.2.1 *Presentation of the Synchronisation*

The objective of this chapter is to introduce CA synchronisation as a state estimator and to evaluate its observational performance. We present this in more detail in the next section. In order to remain consistent throughout the chapter, we use, from now on, the notations of the state estimator, *i.e.* s_t for the state of the system (the driver) and $\hat{s}_t$ for the estimate (the replica). Similarly, the coupling cells between the driver and the replica represent the cells measured by the sensors, thus the set of coupled cells is the set of measured cells $\mathcal{L}_q$. Consequently, the $\hat{s}_t$ state of the replica can be expressed as :

$$\hat{s}_t(c_i) = \begin{cases} s_t(c_i) & \text{if } c_i \in \mathcal{L}_q \\ \hat{s}_t(c_i) & \text{else} \end{cases} \tag{4.1}$$

Before giving the state representation of synchronisation, we shall start with some reformulations of the notions defined in chapter 2. To model the coupling between the two CA, we use a vector representation for the configurations that can easily separate the cells that are coupled and those that are not. To achieve this, we use the vector representation that we have already used in section 2.2.3 for the definition of

ACA and in section 3.2 for the study of the observability and reconstructibility of ACA. This representation provides a vector representation of the s_t configuration where the states of each $s_t(c_i)$ cell is arranged as a vector $x_t = \begin{bmatrix} s_t(c_1) & s_t(c_2) & \cdots & s_t(c_N) \end{bmatrix}'$. Secondly, as we will not be using synchronisation only for ACA then the global transition function F is not necessarily an affine map, hence we shall keep using the notation $x_{t+1} = F(x_t)$ for the evolution of the CA.

The coupling between two CA is done by copying the state of the cells from the driver to the replica. In this case the sensor network strictly respects the definition 2.4 and the output operator is additive and can be expressed with the matrix C_t. However, we will not use the matrix C to represent the coupled cells but the diagonal matrix $P_t = C_t' C_t$. The diagonal matrix P indicates which cells are coupled, the value 1 indicates that the corresponding cells are coupled. Consequently, P_t selects only the coupled cells of x_t whereas $I - P_t$ selects the cells that are not coupled. Finally, the state representation of the synchronisation is expressed as :

Definition 4.1 (Bagnoli *et al.* Synchronisation)**.** Let $\mathcal{A}$ be a CA and F its global transition function. Let $\mathcal{H}$ be a time-dependent sensor network associated to the matrix P_t. The synchronisation of the replica $\hat{x}_t$ over the driver x_t is done by the following:

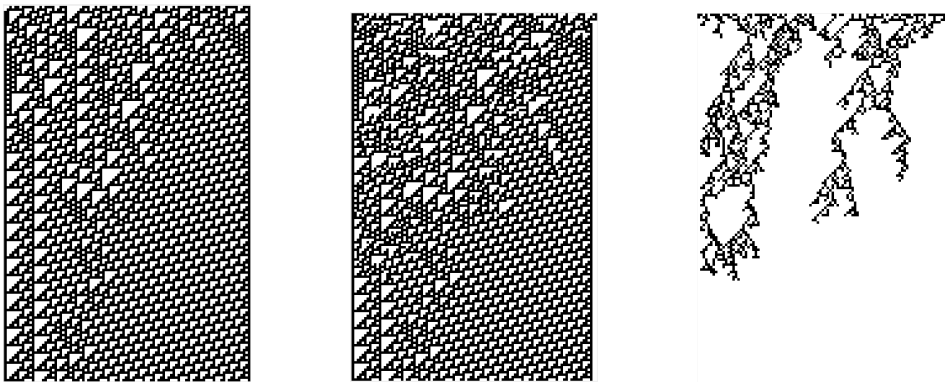
$$\begin{cases} x_{t+1} &= F(x_t) \\ \hat{x}_{t+1} &= (I - P_t) \cdot F(\hat{x}_t) + P_t \cdot F(x_t) \end{cases} \tag{4.2}$$

In (Bagnoli *et al.*, 2010), the coupled cells are chosen randomly at each iteration from a coupling probability p. Different coupling methods exist and are presented in section 4.3. We will study their synchronisation performance in relation to the state estimation problem.

To investigate the synchronisation, we use the synchronisation error e_t defined by eq. (4.3). By doing so, we can represent the difference between the replica and the driver independently of the states of the replica and the driver. Figure 4.1 shows an example of synchronisation for the ECA **R110-100-NB**.

$$e_t = \hat{x}_t - x_t \tag{4.3}$$

The system is considered to be synchronised when $\hat{x} = x$ but it is possible to provide more accurate information by considering the cells that are synchronised $\hat{x}^{c_i} = x^{c_i}$. To do this we define the normalised synchronisation error ϵ_t in definition 4.2 which represents the percentage of cells that are not synchronised at time t. Figure 4.2 describes the evolution of the normalised synchronisation error from the synchronisation example presented in Figure 4.1. This new expression for the



(a) Driver x_t (b) Replica $\hat{x}_t$ (c) Synchronisation Error e_t

FIGURE 4.1. Example of random synchronisation of the ECA **R110-100-NB** for a time horizon $T = 150$ and a coupling probability $p = 0.2$. The black cells represent 1 and the white cells represent 0. Time is going down on the vertical axis.

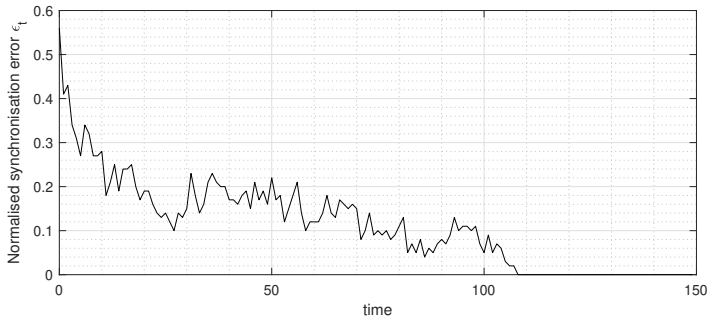


FIGURE 4.2. Evolution of the normalised synchronisation error of the ECA **R110-100-NB** for the example given in Figure 4.1.

synchronisation error makes it easier to represent synchronisation in a temporal perspective in order to analyse the results. It would have been difficult to deduce the time at which the replica is synchronised with the driver from Figure 4.1.

Definition 4.2 (Normalised Synchronisation Error)**.** The percentage of unsynchronised cells within the replica is called the **normalised synchronisation error** ϵ and is express by:

$$\epsilon_t = \frac{1}{N} \sum_{c_i \in \mathcal{L}} \left(1 - \delta\left(e_t^{(c_i)}\right)\right) \tag{4.4}$$

where $\delta\left(e_t^{(c_i)}\right)$ represents the Kronecker delta of $e_t^{(c_i)}$, it is equal to 1 if $e_t^{(c_i)} = 1$ and 0 otherwise.

For the example of ECA synchronisation in Figure 4.1, we have chosen arbitrarily coupling probability $p = 0.2$ and we notice that the replica took about 110 iterations to synchronise to the driver (see Figure 4.2). For the chosen ECA, a coupling probability of 20% represents a total of about 20 coupled cells per iteration. However, we have seen in the previous chapter that not only the position of the sensor is important but also the number of sensors (see example 3.2). Similar concerns can be raised for the synchronisation of the CA. What is the impact of the coupling probability on the synchronisation? Is there a minimal coupling probability that ensures synchronisation?

In order to answer these questions, we shall explore some examples of synchronisation with different coupling probabilities. On Figure 4.3, we can make two hypotheses: First p impacts the synchronisation of the replica (for $p = 0.1$ and $p = 0.2$ the replica does not synchronise and the bigger p is, the faster it synchronises); second, p impacts the normalised synchronisation error decrease (the bigger p is, the fewer black cells at time T). The influence of the coupling probability (number of sensors) on the speed of convergence seems to be consistent with the results of the observation, and in particular with the lower bound of the Cayley-Hamilton Theorem which decreases with the number of sensors. Moreover, after conducting several simulations (not presented here), it seems that the replica never synchronises for $p = 0.1$ and always synchronises for $p = 0.3$ and $p = 0.4$. The case of $p = 0.2$ is more peculiar since the replica synchronises in about one fifth of the simulations in this case.

(a) $p = 0.1$ (b) $p = 0.2$ (c) $p = 0.3$ (d) $p = 0.4$

Figure 4.3. Example of the synchronisation error for different coupling probabilities p for ECA **R110-100-NB**. Time is going down on the vertical axis for 100 time step. The initial configurations of the driver and the replica were randomly initialised but were unchanged for the four examples.

In order to analyse more precisely the impact of the coupling probability on the synchronisation error, we performed a large number of simulations and calculated the mean normalised synchronisation error. This value is represented by the continuous curve in Figure 4.4. Considering the particular case of $p = 0.2$, we notice that $\epsilon = 0.08$. However, as there are some cases where the replica is fully synchronised (*i.e.* $\epsilon = 0$), there is a lack of information to deduce the impact of p on the normalised synchronisation error (apart from the complete synchronisation of the replica). For this purpose, we represent the synchronisation rate, *i.e.* the total synchronisation rate of the replica over all simulations, by the dashed curve.

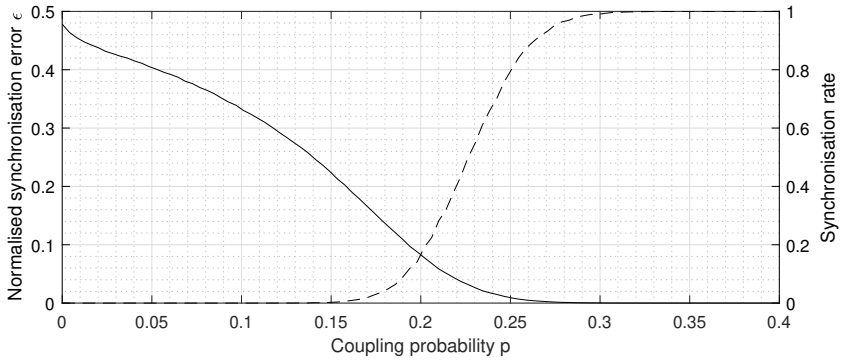


FIGURE 4.4. Evolution of the *normalised synchronisation error* (continuous) and the *synchronisation rate* (dashed) at time $T = 150$ as a function of the coupling probability p for the ECA **R110-100-NB**. The mean is computed through 10^4 simulations.

Thus we notice that the first cases of total synchronisation appear around $p = 0.15$ and the synchronisation rate continues to grow to reach 100% around $p = 0.31$. This confirms our first hypothesis on the direct impact of the coupling probability on the total synchronisation of the replica on the driver. Moreover, by comparing the synchronisation rate with the normalised synchronisation error for probabilities lower than $p = 0.15$, we observe a decrease in the normalised synchronisation error without an increase in the synchronisation rate. For $p = 0.05$, about 40% of the cells are not synchronised while only 22% of the cells are not synchronised for $p = 0.15$. Adding coupling cells brings more information than just the value of the cell itself. We can finally conclude that an increase in the probability decreases the normalised synchronisation error independently of the synchronisation rate confirming our second hypothesis.

In this section we have presented synchronisation and more precisely the synchronisation model of Bagnoli *et al.* (2010). This model is designed to synchronise a replica to a driver by using randomly chosen coupling cells. We have studied an example of synchronisation for the ECA **R110-100-NB** for which we have been able to estimate the critical coupling probability $p_c \approx 0.31$ which permits a perfect synchronisation of the replica on the driver. This perfect synchronisation of the replica on the driver makes it possible to use synchronisation as an observation tool. The state estimator would be the replica that we seek to synchronise with the system in order to determine its state. It is precisely this application of synchronisation that we shall study in the following section.

4.3 Synchronisation as State Estimator

In this section, we present the use of synchronisation as a state estimator. We will start by presenting the connection with state estimation in classical control theory. We will also present how coupling cells can be seen as sensors and how the type of sensor directly impacts the synchronisation performance. We will conclude by presenting some ideas for improving the state estimator for a particular system by using a certain type of sensor or by modifying the state estimator model.

In the case of so-called classical synchronisation, there is not much of a bridge between synchronisation and state estimation because the coupling between the two systems is bidirectional. The state estimator is not supposed to influence the system it observes. In the case of CA synchronisation, it is precisely this unidirectionality of the coupling that makes the link with state estimation possible. Indeed, the state estimator constructs an estimate of the state of the system from measurements made on it, and the estimate is considered "perfect" when the state of the system and the estimator are equal (or at least close enough). We directly relate the state estimator to the replica, the system to the driver and the sensors to the coupling cells.

Moreover, the synchronisation equation (4.2) is similar to the state estimator of linear systems presented in (Kalman, 1960). Indeed, we can compare the expression of the replica of an ACA (4.5) with the expression of the state estimator of LTI systems (4.6). We notice a similar expression where the coupling matrix P represents the product LC (L is the gain matrix which guarantees the convergence of the state estimator) with the difference that the measurements are taken at time $t+1$ for the synchronisation and at time t for the state estimation.

$$\begin{aligned} \text{Synchronisation: } \hat{x}_{t+1} &= (I - P_t) \cdot F(\hat{x}_t) + P_t \cdot F(x_t) \\ &= A\hat{x}_t + P_t(x_{t+1} - \hat{x}_{t+1}) \end{aligned} \tag{4.5}$$

$$\text{State Estimator: } \hat{x}_{t+1} = A\hat{x}_t + LC(x_t - \hat{x}_t) \tag{4.6}$$

Remark 4.1. The gain matrix L is usually computed using a pole placement method (Ackermann, 1972) or a constrained optimisation synthesis (Kalman *et al.*, 1960). These methods could, under the requirement of a proof, be applied to ACA in order to create a state observer for ACA.

As mentioned, the coupling cells represent the sensors of the system. The synchronisation model of Bagnoli *et al.* (2010) takes into account a time-dependent coupling matrix and therefore can handle mobile sensors. However, what is the coupling probability in case of mobile sensors? One can consider that if the cells are randomly observed then it can be considered that they are mobile sensors whose speed of movement is largely superior to the dynamics of the system (sensors can move anywhere in

the lattice at each time). This may also be the situation of a sensor network where only certain sensors are active (as presented in eq. (2.21)). In Figure 4.5, we represented the normalised synchronisation error for random sensors but also for fixed and mobile sensors. The initial position of the sensors is determined randomly in a similar way to the random sensors. In mobile sensor network, their initial position is determined randomly and each sensor moves one cell to the right (as in example 3.1).

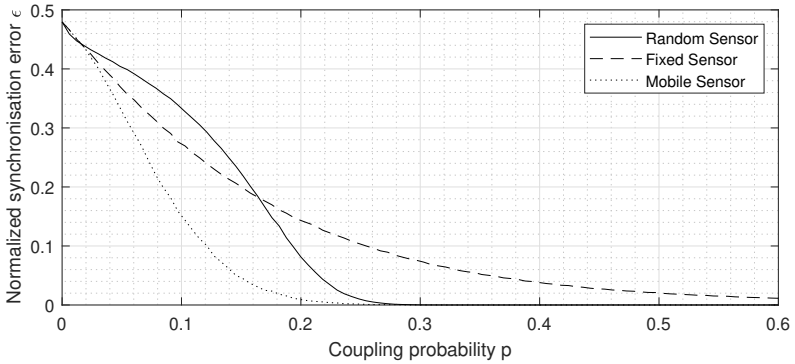


Figure 4.5. Evolution of the *normalised synchronisation error* at time $T = 150$ of the ECA **R110-100-NB** for different types of sensors. The mean is computed through 10^4 simulations.

A notable difference between these three methods is immediately apparent in terms of synchronisation performance. The mobile sensor network performs better than the random sensor network, which itself performs better than the fixed sensor network. These results are, however, very specific to the rule studied. Indeed, if we consider the ECA **R184-100-PB**, which is an example of the road traffic model (see Figure 2.5 and section 5.3), the synchronisation performances represented on Figure 4.6 are quite different. Indeed, in this model the "cars" (cell with state 1) move one cell to the right at each time step. It is therefore impossible for the mobile sensor moving at the same speed to measure the car if it is not on the same cell as the sensor. On the contrary, for the fixed sensor network, as the car is moving on the lattice, it will, sooner or later, be measured and therefore synchronised. Finally, in order to be the least sensitive to this kind of detail during the general studies, we shall stick to random synchronisation. This also allows us to apply the different results of the synchronisation. However, when we apply state estimation with synchronisation in the context of physical system observation (see chapter 5), we will compare in more detail the sensor network topologies as well as the impact of the different coordination methods on the synchronisation performance.

For some applications, in addition to the sensor placement method, the replica model can also be adapted to improve synchronisation performance. The replica model will differ from the driver, which may permit to the replica to converge faster to the driver. In the forest fire example (see section 5.2), the fire spread model is slightly

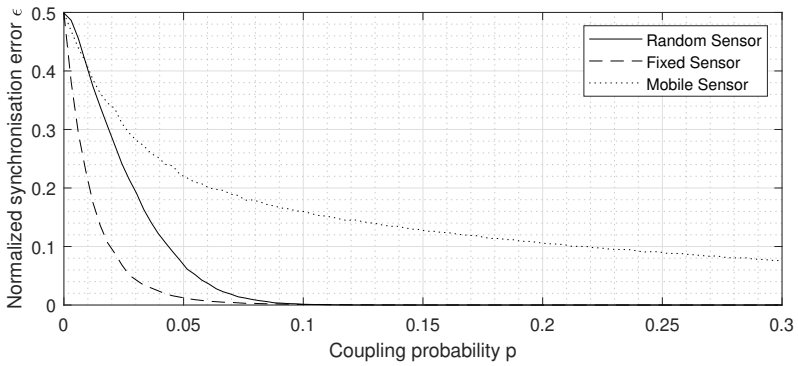


Figure 4.6. Evolution of the *normalised synchronisation error* at time $T = 150$ of the ECA **R184-100-PB** for different types of sensors. The mean is computed through 10^4 simulations.

modified. In the initial model, it is the fire that propagates in the forest, but in the "improved" model, the ashes also propagate. Therefore, when a sensor measures ash, it indicates that a fire was present on that cell some time ago, so the neighbouring cells have also burnt and are now ash. This has the effect that the measurement of a cell with ashes is useful for synchronisation when initially it would not have contributed to reducing the synchronisation error (beyond its own value).

In this section we have seen how the replica of the CA synchronisation can be used as a state estimator. The coupling cells can be considered as sensors. We also observed that sensor density has an influence on the normalised synchronisation error. The sensor network coordination method has an influence on the critical density that guarantees the full synchronisation of the replica. In the following section, we investigate the particular case of a small initial synchronisation error (a few cells of error). The objective will be to model the propagation of the initial error in order to improve the synchronisation algorithm.

4.4 State Estimation Optimisation for Small Initial Error

In this section, we will consider synchronisation when the initial error is small. Indeed, as we stated in the previous chapter, it is likely that for some applications there are constraints on the plausible initial configurations. In this case, the initial synchronisation error is limited. To borrow from the example of the propagation of a forest fire, if the topology of the forest is known and only one cell can be ignited at initialisation, thus there is only one error cell. The synchronisation error is then very small. We will therefore study how this initial error propagates in order to develop an optimised synchronisation method for monitoring systems with a low initial error.

4.4.1 Impact of the Initial Error on the Synchronisation Error

In order to study if and how the initial synchronisation error impacts the evolution of the normalised synchronisation error, we show in Figure 4.7 the normalised synchronisation error ϵ_t as a function of time for different initial error ϵ_0 values. These results were obtained through several synchronisation of the ECA **R18-100-PB** and coupling probability $p = 0.1$. The initial configuration x_0 was randomly initialised and and $\hat{x}_0$ was initialised to the value of the driver with some erroneous cells (1 cell for $\epsilon_0 = 1\%$, 2 cells for $\epsilon_0 = 2\%$ and son on). The initial synchronisation error ϵ_0 has a clear impact on the overall normalised synchronisation error. Its first influence is on the speed of convergence towards the asymptote. Indeed, the 10% curve seems to converge faster than the 20% and 50% curves which converge earlier than the 1% and 2% curves. The second effect of the initial error is on the value of the asymptote when ϵ_0 is small enough. For 10%, 25% and 50%, ϵ_t converge towards the same asymptote value, around 0.23. However if ϵ_0 is sufficiently small, the reached asymptote is lower than this "generic" one.

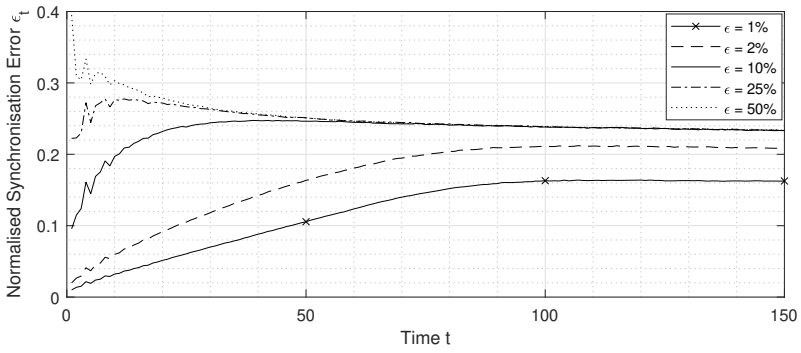


Figure 4.7. Normalised synchronisation error as a function of time of the ECA **R18-100-PB** for different initial error ϵ_0. The mean is computed through 10^4 simulations.

To understand the difference between the two asymptotes, we studied the evolution of the error as a function of time for the particular case of a single cell of initial error ($\epsilon_0 = 1\%$). As we can see on Figure 4.8, there are two very different kind of evolution of the synchronisation error e_t. On one hand, in Figure 4.8a, the error spreads until it covers the whole CA and reaches the asymptotic non-zero value. On the other hand, in the very specific case depicted on Figure 4.8b, the replica synchronise with the driver and the normalised synchronisation error reaches zero. Therefore, when we average these two cases, which we did for Figure 4.7, we obtain a lower asymptotic value. For the remaining of the study, we chose to dissociate the two cases and to not consider the total synchronisation cases when we study the asymptotic value.

(a) Asymptotic Synchronisation (b) Total Synchronisation

FIGURE 4.8. Evolution of the synchronisation error for ECA **R18-100-PB** from a single cell error ($\epsilon_0 = 1\%$). The time is represented on the vertical axis. A black pixel is an erroneous cell in the lattice of the replica.

To characterise the impact of the initial error on the performance of the synchronisation, we only consider the cases of total synchronisation which do not influence the average value. Figure 4.9 represents this asymptotic value as a function of the initial error. First, if we consider only the asymptotic synchronisation (without special cases of early complete synchronisation), the value of the asymptote does not depend on the initial error. Second, the value from which the average error with and without total synchronisation become different depends on the coupling probability p: the stronger, the greater the chance of total synchronisation.

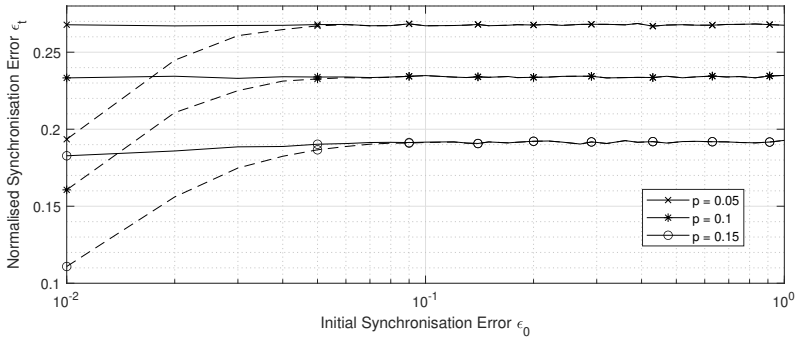


FIGURE 4.9. Value of the asymptotic behaviour of the normalised synchronisation error as a function of initial error ϵ_0 of the ECA **R18-100-PB**. The continuous lines consider only the asymptotic synchronisation whereas the dashed lines include both asymptotic and total synchronisation. The mean is computed through 10^4 simulations.

4.4.2 Modelling of the Error Spreading Dynamics

In order to explain the dynamics of the evolution of the synchronisation error, we study how the error propagates within the CA as a function of the coupling probability. To do so, we start by studying the propagation of the error within the simple case of a single erroneous cell and then, we generalise these results to two or more cells. To study the propagation of the synchronisation error, we shall study the propagation of a single ECA and generalise the results to all ECA. To decide which rule we shall study, we have reported the synchronisation error with null coupling probability in Figure 4.10 as a function of the class of the ECA. Class 1 shows no interest as regardless of the initial configuration, the state of all cells becomes rapidly uniform. Class 2 is also not very interesting because the error does not propagate over the whole lattice. Class 4 seems to be more interesting. However the propagation structure varies strongly depending on the initial configuration. Finally, we will be interested in class 3 but not in the **R30** rule because it is not symmetrical. We eventually choose the **R18** ECA because it is the smallest symmetrical class 3 ECA.

(a) Class 1: **R32-50-PB** (b) Class 2: **R172-50-PB** (c) Class 3: **R30-50-PB** (d) Class 4: **R110-50-PB**

Figure 4.10. Examples of synchronisation error from a single erroneous cell for ECA of different classes. Time is going down on the vertical axis for 25 time step.

Through simulations with the ECA **R18**, we seek to model the propagation of the synchronisation error as a function of the coupling probability p. First, we present on Figure 4.11 the typical synchronisation error propagation dynamics for **R18** from a single erroneous cell.

(a) $p = 0$ (b) $p = 0.05$ (c) $p = 0.10$ (d) $p = 0.15$

Figure 4.11. Evolution of the error for elementary **R18-500-NB** from a single erroneous cell. Time is going down on the vertical axis for 250 time step.

Comparing the different error propagation for the ECA **R18** in Figure 4.11, we adopted a triangle as a simple geometric model for these dynamics. Therefore, to describe how the synchronisation error spreads, two parameters will be used: the first being the **aperture angle** of the propagation triangle, and the second being the **shift angle** between the altitude and the median of the triangle. Indeed, the median

of the triangle seems to vary from one simulation to another. Figure 4.12 describes the geometry associated with these angles α and β which describe respectively the aperture angle and the shift angle. From Figure 4.11, it appears that for **R18** the aperture angle of the triangle is inversely proportional to the coupling probability p and the shift angle is null.

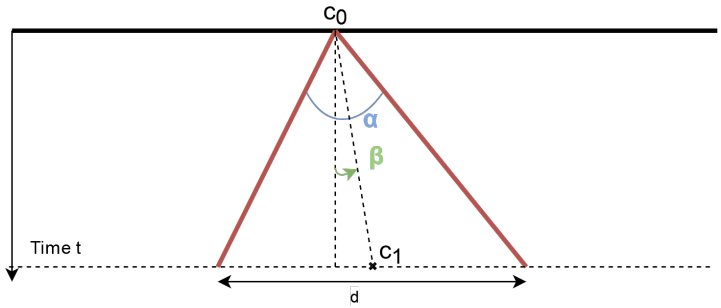


FIGURE 4.12. Schematic of the theoretical propagation triangle of the synchronisation error from a single initial error cell.

For the sake of simplicity, we will not use directly the α and β angles but their tangents, which represents spread velocities (the opposite side is the altitude which is the time here). We will simply call α and β the velocities associated to the angles and not the angles themselves. Therefore, the **error spreading ratio** α represents the mean number of cells by which the triangle base increases at each time step and **error shift ratio** β the mean number of cell shift at each time step. By default, a positive error shift ratio reflects a shift to the right and a negative error shift ratio a shift to the left. As this is the synchronisation of ECA, the error can only propagate, at each time step, a maximum of one cell to the right and one cell to the left. Thus, the coefficient α cannot exceed 2 and the coefficient β cannot exceed 1. The case where $\beta = 1$ can only happen if $\alpha = 0$ because the middle of the base of the triangle is shifting less quickly than its edges (found for example with **R184**).

Based on Figure 4.12, we can express the width d and the middle c_1 of the error at time T by $c_1 = c_0 + \beta \cdot T$ and $d = \alpha \cdot T$. The width d must also be restricted by the size of the lattice N. Subsequently, from the width of the error d, we are able to deduce the normalised synchronisation error ϵ_t from the asymptotic value of the error γ and the proportion of the lattice that is covered by the error d/N. The value of the asymptotic synchronisation error γ is the value of ϵ_t when the timing error is present over the entire lattice. The average value of the asymptote is given in Figure 4.9 for three values of p. We finally propose the following corollary which gives the expression of ϵ_t.

Corollary 4.1. *The normalised synchronisation error ϵ_t can be estimated from the parameter α as well as the value of the asymptote γ associated to the coupling probability p. Thus,*

the normalised synchronisation error ϵ_t is defined by :

$$\epsilon_T = \frac{\gamma}{N} \cdot \min(N;\ \alpha \cdot T)$$

We can calculate the mean value of the error spreading ratio by measuring the area of the error at time T and divide by current time to obtain the tangent of α. Figure 4.13 describes the evolution of the average spread ratio as a function of the coupling probability. For the first part, up to $p = 0.2$, it is similar to an affine function of which we experimentally obtained the equation $\alpha = -8.41 \cdot p + 1.95$ using linear regression. This expression of the error spread ratio supports the hypothesis raised by Figure 4.8 which claims that α is inversely proportional to the coupling probability. As specified in the previous section, we are not interested in cases where synchronisation is total because the propagation of error is zero. However, as we get closer to the critical coupling probability $p_c = 0.3$, the number of total synchronisation increases significantly. The average is therefore no longer made on several thousand simulations but only a few hundred, which implies that the precision of the results decreases as we get closer to the critical coupling probability.

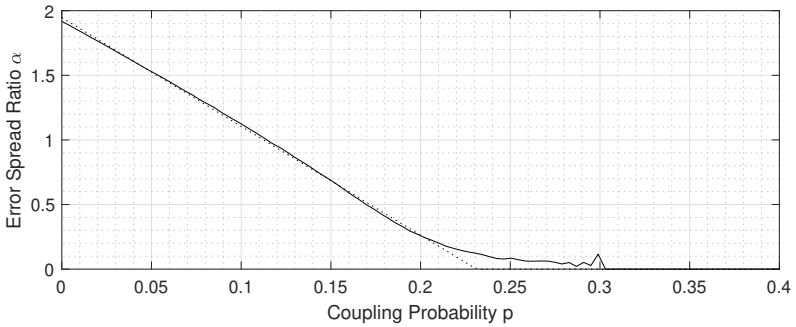
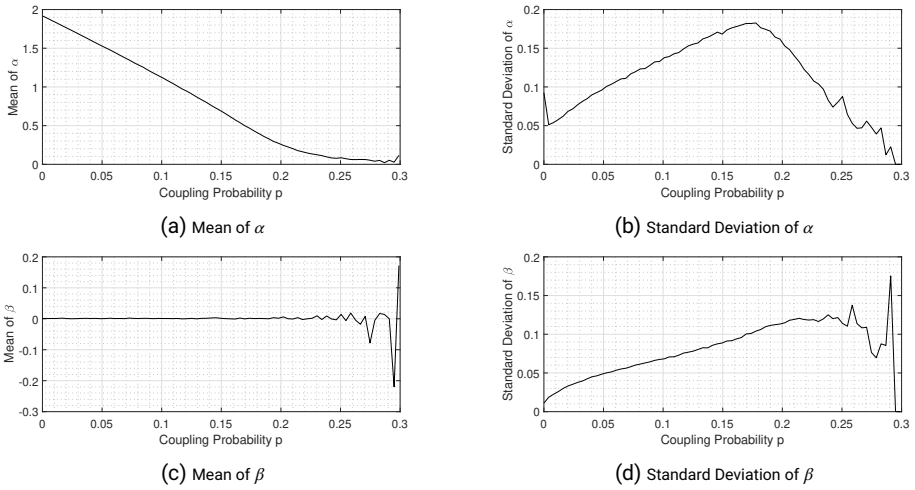


Figure 4.13. The error spread ratio α as a function of coupling probability for the ECA **R18-500-NB**. The continuous curve is obtained by taking the mean of α over 10^4 simulations. The dashed curve is the curve obtained by linear regression.

In Figure 4.13, we have plotted the mean values of alpha as a function of p, but for a given value of p, the α coefficients are normally distributed. The same observation can be made for the error displacement coefficient β. In order to have a better representation of α and β, we can calculate their mean and standard deviation as a function of the coupling probability as we have done for the mean value of α in Figure 4.13. They are shown in Figure 4.14. However, as in Figure 4.13, the number of total synchronisations sharply increases as we approach the critical probability so that the statistics are made with fewer non-zero values.

Using the corollary 4.1, we are going to make an example for $p = 0.1$. For this coupling probability, the error spread ratio α follows a normal distribution with



(a) Mean of α (b) Standard Deviation of α

(c) Mean of β (d) Standard Deviation of β

FIGURE 4.14. Mean and standard deviation of the error spread ratio α and the error shift ratio β as a function of coupling probability for the ECA **R18-500-NB**. The mean and standard deviation were computed over 10^4 simulations.

mean 1.124 and standard deviation 0.136, the asymptotic synchronisation error value taken from Figure 4.9 is approximately $\gamma = 0.221$. From the same figure, we can compute τ, the rate of total synchronisation among all simulation, whose value is $\tau = 0.305$ for $p = 0.1$. It must also be taken into account in order to represent the normalised synchronisation error of Figure 4.7. On Figure 4.15, the mean normalised synchronisation error ϵ_t computed through simulations is displayed as well as the estimated error with an α fixed at the mean and an α that follows the normal distribution. We notice that the use of the normal distribution in the calculation of the error explains the rounded curve when the error approaches the asymptote. However, the two theoretical curves have a difference with the real curve which is explained by a faster increase of the error during the first iterations which is caused by a higher α as the error is not yet detected by the sensor, and therefore unsynchronised.

This method allows us to simply represent the propagation of the error in the case where a single cell is erroneous in the initial configuration. If we consider two or more erroneous cells then the modelling becomes more complex. Indeed, the multiple initial errors propagate independently until they *collide*; as shown in Figure 4.16. Thus, we must consider that the errors merge in a single (larger) source of error. Considering that the collision takes place at time t_1, we can consider that the error spread ratio α for two initial erroneous cells is expressed as:

$$\alpha(t) = \begin{cases} \alpha_0 + \alpha_1 & , \text{if } t \leq t_1 \\ \frac{(\alpha_0 - \beta_0 + \alpha_1 + \beta_1)}{2} & , \text{if } t \geq t_1 \end{cases} \tag{4.7}$$

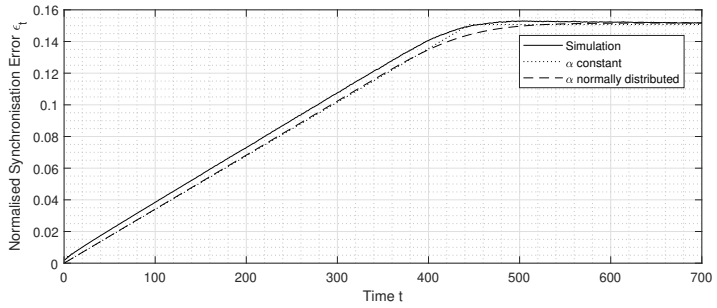


Figure 4.15. Evolution of the normalised synchronisation error as a function of time from simulations of the **R18-500-PB** as well as the two theoretical cases when α is constant or follows a normal distribution.

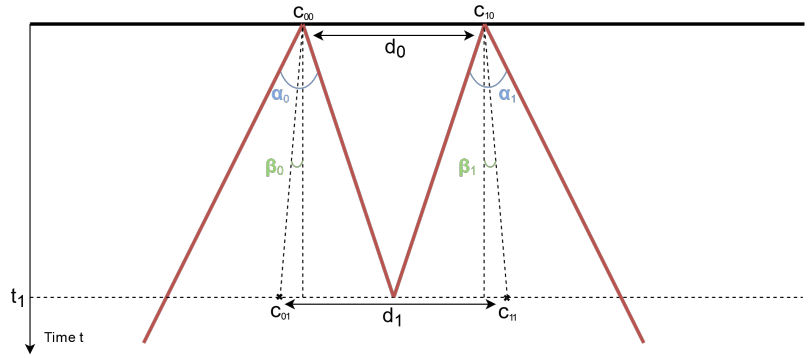


Figure 4.16. Schematic of the theoretical propagation triangle of the synchronisation error from two initial error cells.

The time t_1 of the collision depends on the initial distance between the two initial errors, whose probability distribution depends on the boundary conditions used. Indeed, if we consider that the two erroneous cells are at the two extremities of the lattice, then the distance between the two is either 0 for periodic boundaries or N for the other boundary conditions. Therefore, t_1 can be expressed as :

$$\begin{aligned} \frac{\alpha_0}{2} \cdot t_1 + \frac{\alpha_1}{2} \cdot t_1 = d_1 \iff \frac{\alpha_0 + \alpha_1}{2} \cdot t_1 = d_0 + (\beta_1 - \beta_0) \cdot t_1 \\ \iff t_1 = \frac{d_0}{\beta_0 - \beta_1 + \frac{\alpha_0 + \alpha_1}{2}} \end{aligned} \tag{4.8}$$

Moreover, since each of these initial errors is subject to the total and fast synchronisation (of probability τ), the model must include, with probability $2\tau(1-\tau)$, a propagation with only one initial error using the model of Figure 4.12. With more than two erroneous cells, the operation is the same but it is necessary to take into account several collisions occurring at different times as well as the different scenarios

of fast and total synchronisation. Using the different parameters α, β, γ, and τ used for Figure 4.15, we can do the same thing for two initial error cells. Figure 4.17 shows the three curves: that of the simulations, the constant theoretical value and the values following normal distributions.

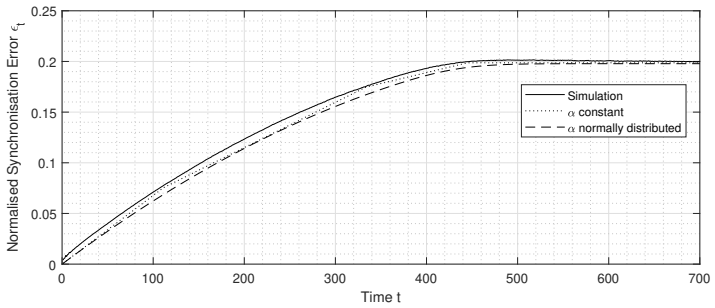


FIGURE 4.17. Evolution of the normalised synchronisation error as a function of time from simulations of the **R18-500-PB** as well as the two theoretical cases when α is constant or follows a normal distribution.

In this section, we first confirm the proportionality between the error propagation speed and the coupling probability. This makes it possible to model the propagation of the synchronisation error for one or two erroneous cells at initialisation. This model could be extended to more erroneous cells, but it becomes more complicated to model beyond several cells. In the next section, we will use this error propagation model to propose a more efficient sensor placement in the case of a single erroneous cell.

4.4.3 *Optimisation of Algorithm for a Single Erroneous Cell*

In this section we propose a method for coordinating sensors for synchronisation where the initial synchronisation error is small. The objective is to identify the cells in the replica lattice that may be in error and then concentrate the sensors in these areas to increase the chances that the error is measured and synchronised. We will first present the construction of the *possible error area* and then adapt the synchronisation of definition 4.1 using this possible error area and compare the performance of the two methods.

In order to identify the error area, the sensor network must first measure an error. From the position of this error cell, the synchronisation error propagation model can be used to find the possible positions of the original error cell. Let us note $\hat{e}_0$ the initial error zone which corresponds to the set of cells that can lead to errors measured at time t. Then, by using the model of the propagation of the synchronisation error, it is possible to find $\hat{e}_t$, the set of cells which can be erroneous at time t by considering $\hat{e}_0$

the initial error zone. Figure 4.18 shows the geometric construction of the $\hat{e}_0$ error zone with a backpropagation of the error measured by the sensor array as well as the $\hat{e}_t$ error zone at time t. The error spread ratio α_{max} corresponds to a ratio α large enough to include all (or a large part) of the possible error spread ratios at the given coupling probability. As α follows a normal distribution, a ratio $\alpha_{max} = \alpha_{mean} + 3\sigma$ encompasses 99.7% of the possible spreading ratios. However, as we have seen in Figure 4.15, the error spread ratio is larger at the beginning since the error is still too low to be synchronised. We must then consider the maximum ratio for this case, *i.e.* $\alpha_{\max} = 2$.

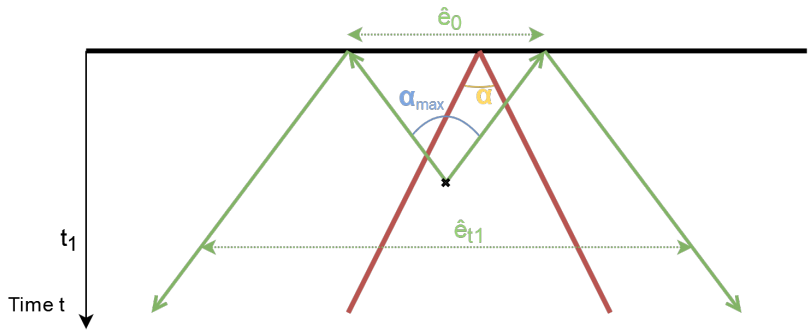


Figure 4.18. Schematic of the backpropagation of the synchronisation error to find the initial error area $\hat{e}_0$

In a formal sense, we define the initial error zone $\hat{e}_0$ corresponding to the erroneous cell c_i at time t as:

$$\Xi_{\alpha,t} : c_i \mapsto \hat{e}_0 = \{c \in \mathcal{L} | \ |c - c_i| \leq \frac{\alpha \cdot t}{2}\} \tag{4.9}$$

Similarly, we define the error zone $\hat{e}_t$ at time t from $\hat{e}_0$ by :

$$\Phi_{\alpha,t} : \hat{e}_0 \mapsto \hat{e}_t = \{c \in \mathcal{L} | \ \exists \ c_i \in \hat{e}_0, |c - c_i| \leq \frac{\alpha \cdot t}{2}\} \tag{4.10}$$

As new errors are detected by the sensor network, we can adapt the initial error zone. Indeed, as we know that there is only one erroneous initial cell, we know that this cell is in every $\hat{e}_0$. Therefore the initial error area can be refined using the intersection of all the initial error areas of all the errors detected by the sensors. In this way, it is possible to reduce the size of the error zone at time t but also to locate the position of the initial error. We can express the evolution of this by $\hat{e}_0^{(t)}$ (4.11) where $\mathcal{Y}_t$ represents the set of erroneous cells measured by the sensor network at time t.

$$\begin{cases} \hat{e}_0^{(t+1)} & = \hat{e}_0^{(t)} \cap \bigcap_{c \in \mathcal{Y}_t} \Xi_{\alpha,t}(c) \\ \hat{e}_t & = \Phi_{\alpha,t}(\hat{e}_0^{(t)}) \end{cases} \tag{4.11}$$

Now that the error area can be estimated, it remains to position the sensors. The method consists in placing the sensors only in the area where the error could be present. The sensors are still placed randomly based on the coupling probability. As the area of observation is reduced, we can increase the probability of coupling in proportion to the reduction of the area of observation. Thus, the density over the whole lattice remains the same but the density over the error area increases. This new method will result in a lower critical coupling probability p_c as shown in Figure 4.19. The coupling probability in the error area $\hat{e}_t$ is described by:

$$p_{error} = p \cdot \frac{N}{|\hat{e}_t|}$$

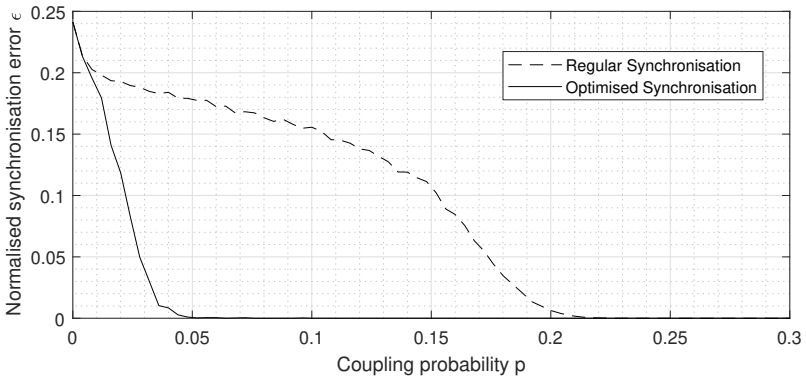
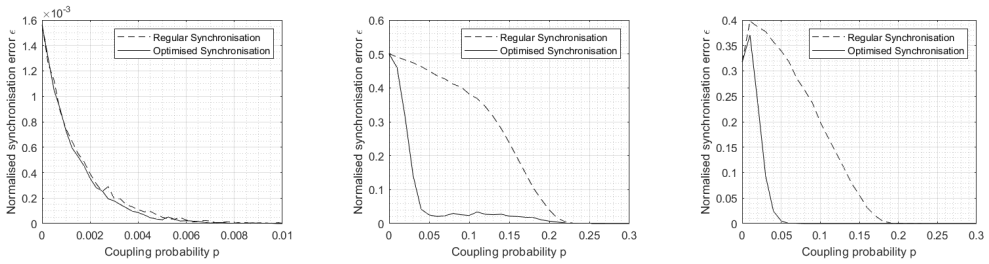


FIGURE 4.19. Evolution of the normalised synchronisation error at time $T = 750$ of the ECA **R18-500-PB** for the regular and the optimised synchronisation. The mean is computed through 10^3 simulations.

As shown in Figure 4.19, the optimised synchronisation performs better than the usual one with a critical control strength p_c at 0.05 instead of 0.21. However, when the control is too weak, the difference between the two is negligible because the first error cell is detected too late by the sensors. The error area $\hat{e}_t$ is too large and therefore the optimised coupling probability p_{error} is not sufficient to synchronise the replica on the driver.

In Figure 4.20, we have compared the optimised synchronisation to the regular one for the ECA belonging to different classes presented in Figure 4.10. The results obtained by using the maximum error spread ratio used for the backpropagation $\alpha_{max} = 2$. In the three examples, the optimised synchronisation performs better than the regular one. **R172-500-PB**, and more generally Class 2 ECA, exhibit a lower critical coupling probability due to the fact that the error does not spread on the whole lattice and remains constant after a few iterations. Because the error is only located on certain cells of the CA, the synchronisation time directly affects the synchronisation performance. Indeed, if it is synchronised for a sufficiently long time,

then all the error cells have been coupled and therefore synchronised in a random manner. In this case, the synchronisation performance depends strongly on the synchronisation time horizon and less on the synchronisation method (classical or optimised synchronisation). Therefore, using the optimised synchronisation results in a lower (or even insignificant) performance gain than for the other examples. The use of optimised synchronisation with **R30-500-PB** does reduce the normalised synchronisation error, but not the critical coupling probability. This is due to the fact that the use of optimised synchronisation allows the replica to be synchronised in a large number of cases, but it is necessary to reach the critical coupling probability $p \approx 0.3$ in order for the complete synchronisation to occur. The average is therefore lower but, as in Figure 4.7, it is an average between synchronised cases and unsynchronised cases at the asymptote.



(a) Class 2: **R172-500-PB** (b) Class 3: **R30-500-PB** (c) Class 4: **R110-500-PB**

Figure 4.20. Evolution of the normalised synchronisation error at time $T = 750$ for different ECA for the regular and the optimised synchronisation. The mean is computed through 10^3 simulations.

The presented sensor placement method only works for one error cell. If the second error cell is not in $\hat{e}_0$ after the first measurement, then no sensor will be able to measure that error because the probability outside the error area is 0. It is therefore necessary to consider two initial error zones $\hat{e}_0^{(1)}$ and $\hat{e}_0^{(2)}$ for the two error cells. If the sensors measure an error cell that does not correspond to one, it is the other. Thus, if the errors are sufficiently distant at the beginning then it is possible to have two disjoint error zones. Otherwise it is more difficult to know which error triangle the measured error cell belongs to.

Several ideas could be put in place to deal with this case. For example, Plénet *et al.* (2020b) use sensors with different roles to monitor the propagation of forest fires. The fire model is represented by a CA but they do not use synchronisation, the replica does not have the same dynamics as the driver. In this example, some of the sensors called "followers" have the objective to measure the cells in the error zone while the others, called "explorers", measure the outside of the zone in order to detect new ignitions. In the case of CA synchronisation, in the error zones, the trackers have a sufficiently high coupling probability to guarantee synchronisation and outside the zone the explorers have a lower probability but sufficient to "detect" new errors.

4.5 Conclusion

In this chapter we have studied the synchronisation of cellular automata. We have first seen how it can be used as a state estimator and especially that its definition is very close to the state estimators of classical control theory. Next, we studied the synchronisation for the particular case where the initial synchronisation error is very small. We were able to model the propagation of this initial error using basic geometry in the case of one or two error cells. We ended this chapter by proposing a sensor placement method that uses this model to determine the error area. This method of sensor placement is more efficient than the usual synchronisation and in some cases allows a lower critical coupling probability to be obtained.

Finally, synchronisation appears to be a good alternative to the notions of observability and reconstructibility when the studied CA have a large number of cells. However, the study of the performances provides a statistical estimation of the observation performances but does not guarantee, as opposed to the reconstructibility, the correct observation of the system. In the next chapter, we shall illustrate synchronisation, observability and reconstructibility on practical examples and will be able to compare in detail the synchronisation with the approaches proposed in chapter 3.

Chapter 5

Application to Monitoring of Complex Systems

Résumé

Dans ce dernier chapitre, nous allons appliquer les méthodes d'observation présentées dans les chapitres précédents sur trois exemples concrets. Pour le premier exemple, nous allons étudier un modèle de propagation de feu de forêt. Cet exemple permettra de détailler la méthode de construction d'un estimateur d'état en utilisant à la fois l'observabilité d'AC non linéaire et la synchronisation. Dans le second exemple, nous étudierons un modèle de trafic routier et plus particulièrement le problème de la reconstructibilité décentralisée permettant de construire un observateur d'état modulaire pour la surveillance des réseaux routiers. Nous terminerons avec un exemple d'AC générateur de nombres aléatoires. Nous utiliserons l'observabilité des AC affines pour trouver les nombres aléatoires générés par celui-ci.

Contents

5.1 Introduction

The main objective of this last chapter is to illustrate the observability assessment methods and tools presented so far by applying them to three very different examples. The aim is not to solve these systems' observation problems but to clarify our approach.

For the first example, we will study a forest fire propagation model. The objective will be to set up a method to detect and monitor the fire spread in an efficient way. To do this, we will first use a relatively simple fire spread model and compare it with a more complex model. We will then detail the method of constructing a state estimator using both non-linear CA observability and *synchronisation*. We will then conclude this example by comparing the advantages and drawbacks of both methods.

In the second example, we will study a road traffic model. We will first study its reconstructibility and then we will study the decentralised reconstructibility. We will finish by studying the reconstructibility of an intersection model with the objective of building a modular state estimator for larger road network monitoring.

For the last example, we will study a CA-based random number generator. We will use the observability of affine CA to find the series of random numbers generated by it from a small number of observations. We will show how observability can be used in a hardware attack.

5.2 Detection and Monitoring of Forest Fire Spread

In this first example, we will study a forest fire propagation model with the intention of providing forest fire detection and monitoring. We will start by choosing a fire propagation model that corresponds to the definition 2.1 for CA, that we made at the beginning of chapter 2. Then, we will present the different parameters that the model can take into account. Afterwards, we will study the observability, the reconstructibility, and the synchronisation of this model in order to build a state estimator. This example will serve for the comparison between the two different methods we have studied in this thesis.

5.2.1 *Presentation of the fire propagation model*

In the introduction to this thesis, we discussed two fire propagation models, the first proposed by Karafyllidis and Thanailakis (1997) which uses the percentage of biomass consumed as a state and the second, a trivial fire propagation model which has discrete states to represent the fire. However, as we pointed out at the end of

section 2.2.2, the cell state of the Karafyllidis and Thanailakis model is continuous and cannot be represented with the definition of cellular automata by El Yacoubi. Therefore, the results on observability and synchronisation can only be applied to the second model.

In this section, we first present the Karafyllidis and Thanailakis model by highlighting the different aspects it can model. Then, we will define, in relation to definition 2.1, the trivial fire propagation model and we will demonstrate that this trivial model can be used to model some of the propagation dynamics present in the Karafyllidis and Thanailakis model. Finally, we will present several properties of the trivial fire propagation model, such as percolation, which is essential to understand some results on observability and synchronisation.

Karafyllidis and Thanailakis Model

Karafyllidis and Thanailakis (1997) developed a CA model of forest fire propagation that is able to model the circular propagation of forest fire and to take into account some external factors such as the wind or the terrain elevation. The model makes use of a regular lattice of square cells where the cell state represents the percentage of biomass consumed by the fire. The state varies continuously between 0 and 1, where 0 represents an unburnt forest square and 1 a fully burnt one.

To spread the fire, the authors chose to define the consumed biomass at time t as the sum of the consumed biomass of the neighbours, with factors that depend on different parameters presented in the next paragraph. This method is very similar to the one we use for Additive CA (ACA). However, the CA of Karafyllidis and Thanailakis is not an ACA due to the saturation of the cell state. Indeed, the consumed biomass is defined between 0 and 1; therefore the transition function uses saturation arithmetic between 0 and 1. Consequently, by considering the local transition function (2.8) and adding the saturation, we obtain the following local transition function where the a_j represents the weight values for the neighbourhood cells:

$$s_{t+1}(c_i) = \min\left(1;\ \sum_{j=1}^{n} a_j \cdot s_t(c_i^{(j)})\right) \tag{5.1}$$

In this CA, the Moore neighbourhood is used. When the coefficients for all the cells are equal, the fire propagation front has a square shape (see Figures 5.5a and 5.5b). To solve this problem and have a circular fire propagation front, the authors used a Moore neighbourhood with lower weights in the 4 corners. In this way, the neighbourhood offers an "octogonal" shape (see Figure 5.1a) because the corners are only covered at 83%. We denote this parameter A_j which is equal to 0.83 if the cell c_j is in the corner of the neighbourhood and 1 otherwise. The parameters related to the wind and the terrain topography are directly managed in the neighbourhood

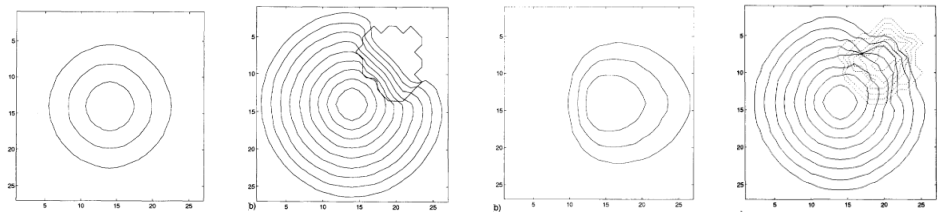
(a) Octagonal Moore Neighbourhood (b) Wind-Elevation Neighbourhood

FIGURE 5.1. Definition of the neighbourhoods for the Karafyllidis and Thanailakis fire propagation model.

coefficients. For example, a north wind accelerates the propagation of the fire towards the south, on the neighbourhood. This is transcribed by a stronger coefficient on the cell located in the north than the other cells. Topography is modelled in the same way, higher grounds have higher coefficients. Figure 5.1b shows the coefficients and their position in the neighbourhood, H_j represents the coefficient related to the topography on cell j and W_j represents the coefficient related to the wind on cell j. As the topography varies with the position of the cell (this can also be the case for the wind), the coefficients of the neighbourhood change depending on the cell, so the CA is hybrid. The last parameter taken into account is the burning rate of the plant in the cell. In the same way as the topography, this changes according to the cell but applies uniformly to all the neighbours of the target cell. Karafyllidis and Thanailakis defined this value by $R = a/t$, where R is the burning speed m/s, t is the burning time (s) and a is the cell length (m). Finally, given the neighbourhood coefficient A, R, W and H, equation (5.1) can be written as :

$$s_{t+1}(c_i) = \min\left(1;\ s_t(c_i) + R \cdot \sum_{j=2}^{9} A_j \cdot H_j \cdot W_j \cdot s_t(c_i^{(j)})\right) \tag{5.2}$$

The impact of these four parameters can be seen in Figure 5.2 where the fire fronts are represented in four different scenarios highlighting each of the parameters.



(a) Homogeneous flat forest (b) Heterogeneous flat forest (c) Homogeneous flat forest with a west-east wind (d) Homogeneous forest with elevation

FIGURE 5.2. Successive fire fronts for different scenarios. The unit on both axes are arbitrary. Image taken directly from (Karafyllidis and Thanailakis, 1997).

While capturing some of the important parameters of fire propagation, the model proposed by Karafyllidis and Thanailakis (1997) has some flaws. Several other authors have improved the model. Encinas *et al.* (2007a) have slightly modified the neighbourhood construction in order to provide a circular propagation of the forest fire. In (Encinas *et al.*, 2007b), they also adapted the CA with a hexagonal cell mesh. Berjak and Hearne (2002) added the moisture factor and validated it on a savannah fire in South Africa. Although there are many variants of this model, it is fundamentally a deterministic model which is not able to model certain factors such as new fire starts due to firebrands (Perryman *et al.*, 2012).

In the next section, we will present a simple fire propagation model which respects definition 2.1 and for which we will study observability and synchronisation. We could have quantified the continuous states of this CA into k different states, similarly to what is done in fuzzy logic, where the state $a \in \mathcal{S}$ represents the percentage $a/(k-1)$ of consumed biomass. However, this model would not have been as accurate as the Karafyllidis and Thanailakis model anyway. Instead, we chose to study a discrete and trivial model with only 4 states: *empty*, *tree*, *burning tree* and *burnt tree*. We will see that even if its formulation is trivial, it is still a model that can deal with different factors such as the circular propagation of the fire or wind, despite the fact that it has less granularity than the continuous model of Karafyllidis and Thanailakis.

Trivial Fire Propagation Model

What we will call the trivial fire propagation model is a discrete-state model where the cells can assume the following four states: *empty*, *tree*, *burning tree*, and *burnt tree*. The local transition function may vary from one application to another and from one type of model to another. Indeed, this model has already been studied many times, both as a deterministic model (Alexandridis *et al.*, 2011) and as a probabilistic one (Almeida and Macau, 2011). In our example, the transition function is deterministic and is represented by the diagram in Figure 5.3. A *tree* only catches fire when at least one of its neighbours (a discussion about the neighbourhood will follow further on) is on fire. In this case, the cell remains in the *burning tree* state for one iteration and becomes a *burnt tree*. The *empty* and *burnt tree* states are absorbing as well as the *tree* state as long as there is no fire.

In order to be able to apply the results we developed for observability and reconstructibility, we will formulate this CA according to definition 2.1. Therefore, we will consider a mesh of $N \times M$ square cells as well as a state space with the integer values 0, 1, 2, and 3 which correspond respectively to the state: *empty*, *tree*, *burning tree* and *burnt tree*. Formally, the parameters $\mathcal{L}$, $\mathcal{S}$ and f can then be formulated as :

- $\mathcal{L} = \{0, 1, \ldots, N-1\} \times \{0, 1, \ldots, M-1\}$

- $\mathcal{S} = \{0, 1, 2, 3\}$
- $f: s_{t+1}(c_{i,j}) = \begin{cases} 0 & = \text{if } s_t(c_{i,j}) = 0 \\ 1 & = \text{if } s_t(c_{i,j}) = 1 \text{ and } \nexists\, c \in \mathcal{N}(c_{i,j}), s_t(c) = 2 \\ 2 & = \text{if } s_t(c_{i,j}) = 1 \text{ and } \exists\, c \in \mathcal{N}(c_{i,j}), s_t(c) = 2 \\ 3 & = \text{if } s_t(c_{i,j}) = 3 \text{ or } s_t(c_{i,j}) = 2 \end{cases}$

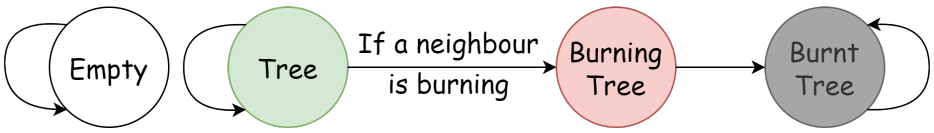


FIGURE 5.3. Diagram of the state changes of the local transition function of the trivial fire propagation model.

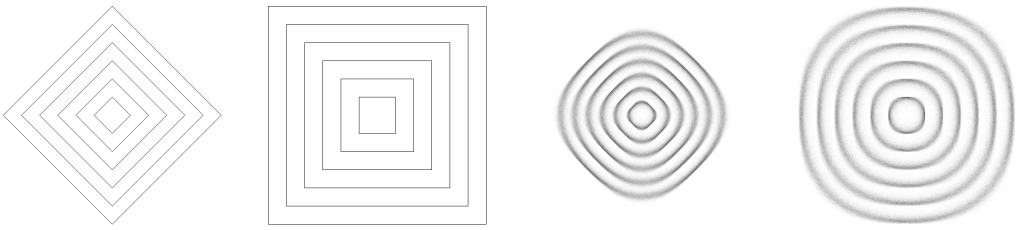
The choice of the neighbourhood provides the possibility to change the shape of the fire propagation and consequently to perform a circular propagation as it is the "natural" propagation shape. For their fire propagation model, Karafyllidis and Thanailakis used a Moore neighbourhood with lower coefficients for the angles. While using discrete states, we cannot use this artefact and the fire propagation shapes for the Von Neumann and Moore neighbourhood are far from circular (see Figures 5.5a and 5.5b). For this reason, we decrease the density of the forest. At the initialisation of the forest not all cells will be trees, but some will be empty. The density is defined as the probability for a cell to be initiated as a tree cell. For example, with a density $d = 0.7$ each cell has a probability $p = 0.7$ of being a tree and a probability $p = 0.3$ of being empty. Therefore, by decreasing the density sufficiently, the fire cannot propagate in a straight line (see Figure 5.4). When taking the mean for several fire spreads (or when considering a much larger number of cells), the mean fire front becomes circular.

(a) $t = 0$ (b) $t = 10$ (c) $t = 20$ (d) $t = 30$

FIGURE 5.4. Example of the evolution of the trivial CA of fire propagation for a Moore neighbourhood with density $d = 0.5$. The CA used is of size 50×50 and state snapshots are taken every 10 iterations.

In order to have a circular shape, we use a density $d = 0.7$ for the Von Neumann and a density $d = 0.5$ for the Moore neighbourhoods (see Figures 5.5c and 5.5d). A

lower density can be used for the Moore neighbourhood because the neighbourhood contains more cells. The choice for these density values are discussed in the next section when studying the percolation of the system. The density cannot be decreased below the percolation threshold because the fire would not propagate anymore. For the considered fire propagation model, we will use the Moore neighbourhood with the density of 0.5. As both neighbourhoods offer circular propagation, we prefer to use the Moore neighbourhood as it will allow a wider range of wind directions (see next paragraph).



(a) Von Neumann and $d = 1$ (b) Moore and $d = 1$ (c) Von Neumann and $d = 0.7$ (d) Moore and $d = 0.5$

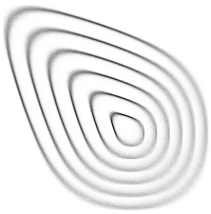
Figure 5.5. Example of the mean evolution of the fire front as a function of the chosen neighbourhood and the tree density at initialisation. Means are obtained over 1000 simulations.

To model the impact of wind on fire propagation, Karafyllidis and Thanailakis (1997) use different coefficient values for the neighbourhood cells. As already mentioned, this is impossible with the trivial model. Therefore, we will enlarge the neighbourhood in the wind direction and thus accelerate the propagation speed in one direction only. This method has already been studied by Trunfio *et al.* (2011) where the shape, size and direction of the neighbourhood is the main feature to model the progression of the forest fire. The new neighbourhood is constructed like the previous one with an additional cell depending on the wind direction. Figure 5.6a shows the position of this additional cell in the neighbourhood as a function of wind direction. Figures 5.6b and 5.6c show the progression of the fire with a northwest or east wind. The wind direction could change over time and space, but then the hybrid CA formulation presented in section 2.2.3 should be used.

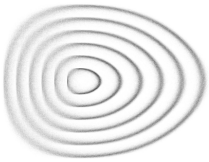
With this model it is difficult to model the heterogeneity of the forest or the topography of the terrain. These two notions require a greater finesse in the speed of fire propagation. While remaining discrete, we have the possibility to deepen the model by lengthening the transient phase, *i.e.* the burning phase of the tree. This would provide the ability to model trees with different burning times t_c. In their work, Schlotterbeck *et al.* (2016) use three different plants with different combustion period: *tree* ($t_c = 200$), *shrub* ($t_c = 70$) and *grass* ($t_c = 1$). This increase in the transition period can be modelled in different ways. The first, which respects definition 2.1, consists in adding intermediate states (*e.g. burning tree 1* to *burning tree 200*). The second requires the use of memory-based CA, a variant of CA where the next state

nw	*nnw*	*n*	*nne*	*ne*
wnw	1	1	1	*ene*
w	1	1	1	*e*
wsw	1	1	1	*ese*
sw	*ssw*	*s*	*sse*	*se*

(a) $\mathcal{N}$ depending on wind direction



(b) Northwest wind



(c) East wind

Figure 5.6. Example of the evolution of the fire front according to the wind direction. Simulations are obtained through the mean over 1000 simulations.

s_t is computed from the n previous states (s_{t-1}, s_{t-n}). This method does not add an extra state, but changes the formulation of the local transition function. For example, the condition of the fourth line of the transition function of the trivial model with a combustion period t_c becomes :

$$\text{if } s_t(c_{i,j}) = 3 \text{ or } \forall\ 0 \leq i < t_c, s_{t-i}(c_{i,j}) = 2 \text{ and } s_{t-t_c}(c_{i,j}) = 1$$

The trivial model we just presented provides a circular propagation of the fire while taking wind into consideration. Although there are several ways to improve the model, taking into account several parameters necessary for the good modelling of the propagation of a fire, it remains rather unconfigurable. Nevertheless, it will provide us with the opportunity to study the observability and synchronisation for a simplistic fire propagation model that captures some of the important properties of fire propagation. In sections 5.2.2 and 5.2.3, we will focus on windless propagation in a flat, homogeneous forest. In the next section, we will present two properties of this model related to percolation and Wolfram class.

Properties of the Model

In this part we will study the propagation properties of the fire in this CA and how it directly impacts the observation of it. We will start by studying the percolation, *i.e.* the way the fire propagates as a function of the forest density. Then we will discuss small and very large isolated forest areas and we will finish with a short study on the class of this CA in relation to the Wolfram classes defined in section 2.2.4. These different properties need to be studied before carrying out the study of observability, reconstructibility, and synchronisation as they provide essential elements to understand the behaviour of the system. Sensors must be correctly placed according to the dynamic of the system in order to efficiently measure the system.

Percolation initially refers to a fluid passing through a porous material. In our case, and more generally in complex systems, it is a question of finding out whether a

disturbance in the system can spread throughout the system. For a forest fire, it is a matter of determining whether a fire in a given place can ignite the whole forest (or at least a very large part of it). The percolation is directly related to the density of the forest. If the density is too low, then the trees are too far apart and the fire does not spread. On the opposite, if the whole system is composed of trees, then the fire will have no difficulty in spreading over the whole lattice. To evaluate the percolation of the system, the forest is randomly initialised with a fixed density and one and only one tree is ignited. The fire spreads until there are no more fire cells,then we measure the burning tree rate, *i.e.* the percentage of trees that have burnt. Figure 5.7 shows the percolation for the trivial fire model with the two neighbourhoods: Moore and Von Neumann. The two curves, although having different values, are both sigmoid with similar slopes. They both have a very slow, almost stable, growth phase at the beginning and end. In between appears a phase transition of very fast growth. As Moore's neighbourhood is larger, the fire has an improved ability to spread (especially using the diagonals). This distinction between the two neighbourhoods explains the difference between the two curves.

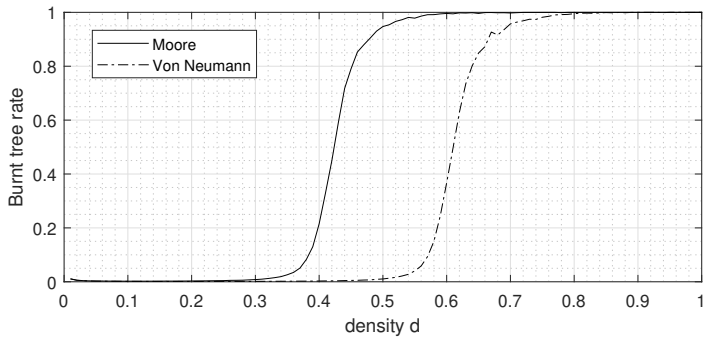


FIGURE 5.7. Percolation of the trivial CA as a function of the neighbourhood. The simulations were performed on a CA of size 100×100 with an average over 1000 simulations.

Earlier we mentioned specific density values to obtain a circular propagation. These values were chosen to get a burnt rate of 95%. This means that the vast majority of the forest burns, but some parts remain isolated and will only ignite when the fire starts in those parts. These isolated forest areas have a number of cells that are distributed exponentially with parameter $\lambda \approx 0.7$. This means that more than half of these areas are composed of only one cell. The fact that there are isolated forest areas makes it more difficult to verify reconstructibility. Indeed, if these areas did not exist, the whole forest would be burnt whatever the starting point of the fire is. The final state s_t for which the whole forest is burnt is the same for all initial conditions. Therefore the reconstructibility is trivial with one and only one final configuration. In this case, the objective of a finer reconstructibility analysis would be to ensure

it as soon as possible, before the fire has spread to the whole system. Whatsoever, the existence of isolated zones avoids this particular case of trivial reconstructibility. Note that this does not impact in any way the study of observability, as we are then interested in the initial state, not the current one.

The objective in this example is to monitor the spread of the fire. It is therefore necessary to reconstruct the s_t state of the system before the forest has burnt down. In order to know the maximum time horizon for the study of observability and reconstructibility, we first need to estimate the average time of the fire propagation. Using the same simulation set as in Figure 5.7, *i.e.* a 100×100 cell lattice, with a density of $d = 0.5$, we find that the mean burning time is 108 iterations with a standard deviation of 23 iterations.

Finally, with the previous results we can quickly deduce that the trivial CA we are studying is of class 2. That is, a CA that reaches a fixed or oscillating state (more details in section 2.2.4). Indeed, after a certain time, the system reaches a heterogeneous steady state where no cell is on fire anymore. However, the error may spread over the whole system before stabilising (if the density is high enough). In this case, the "filter" aspect of class 2 cannot be applied. As a result, the findings of the synchronisation on class 2 CA should be treated with caution as the error spread coefficient will not be as small as it should be. Additionally, there are two special cases (for densities $d = 0$ and $d = 1$) for which the state of all cells is the same at the end of the simulation. For these two cases, the CA is class 1.

In this section we have been able to study different properties that will directly impact the study of observability or synchronisation. The placement of the sensors will be a crucial element in order to measure all the "small" isolated forest areas. However, this CA has a relatively simple dynamic that seems to facilitate its study. In the next two sections we will build, or at least provide the tools to build, a forest fire monitoring system in order to know the state of the system as soon as possible, and especially before the forest has completely burnt down.

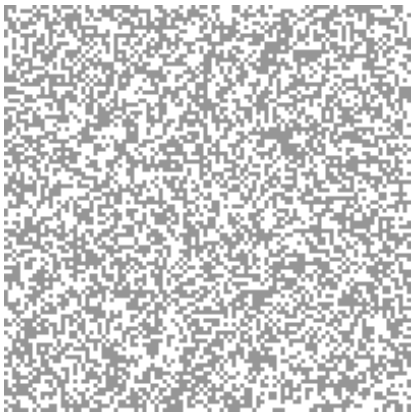
5.2.2 *Observability*

In this section we will study the observability and reconstructibility of the trivial CA model for the forest fire propagation. We will focus on a CA of size 100×100. In order to perform the study in a feasible time, we set conditions on the initial configurations that the system can take (as stated in section 3.3.2). At initialisation, there will be at most one cell that is in the *burning tree* state, all others will be either in the *tree* or *empty* state. We will also notice that ensuring the observability or reconstructibility of this system is not an easy task. That is why the objective of this example is not to deliver a sensor network capable of observing the forest, but only to demonstrate that

the tools proposed in this thesis can be used for the monitoring of the propagation of a forest fire.

The study of observability and reconstructibility for this system is not easy. Indeed, the system is made up of several disconnected areas of forest, the largest of which contains on average 95% of the trees and the smallest, present in very large numbers, have only one or two cells. In order to verify observability and reconstructibility, it will be necessary to be able to determine in which zone the fire is located. To do this, sensors will have to be placed in each of these areas. Therefore, a large number of sensors will be needed, firstly to ensure the observability of the largest area, but also to ensure the observability in all the smaller areas.

The study of observability and reconstructibility will be carried out on a single forest topology. We choose the topology presented in Figure 5.8a which is a 100×100 cell lattice initialised with a density $d = 0.5$. This forest generated has a total of 5054 trees distributed in a total of 50 distinct zones (see Figure 5.8b for more details on the sizes of these zones). In order to limit the number of operations (detailed in section 3.3.2) to evaluate observability, we will limit ourselves to the study of a single fire outbreak. At initialisation, there will be a maximum of one cell that is on fire. As a result, the observability and reconstructibility study will be carried out on 5055 initial configurations: 5054 configurations for all possible fire outbreaks and 1 configuration without any fire outbreak.



(a) Forest topology

Area size	Quantity
1	28
2	6
3	3
4	6
6	1
7	2
8	1
16	1
20	1
4917	1

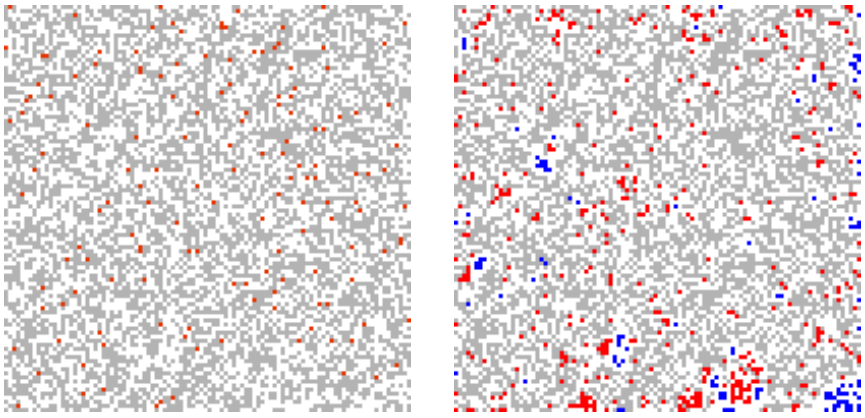
(b) Size and quantity of forest areas

Figure 5.8. Topology of the forest that will be used throughout this example. The forest is of size 100×100 and built with density $d = 0.5$.

As stated at the beginning of this section, we want to know the state of the system in order to monitor the spread of the fire. Thus, if the time horizon required for observability is too large, the state estimation is useless. In the previous section, we estimated the average time of the fire at 108 iterations for a density $d = 0.5$.

We will choose a time horizon $T = 30$, this choice is arbitrary and can be adapted according to the purpose of the monitoring. To perform the observability analysis, we need to choose first the sensor network. The choice of the position or trajectory for the sensors in the network remains an open problem (and not very well studied in the case of mobile sensors). The aim of this example is not to give a solution for forest fire monitoring, but to show that the methods presented in this thesis can be important tools for fire monitoring. In this sense, we will use a network of 150 static sensors placed randomly (see Figure 5.9a) on the trees in the forest (more details on the credibility of this type of sensor in remark 5.1). We will discuss possible sensor placement methods with more details towards the end of this section.

Remark 5.1. The choice of this kind of sensor is not necessarily irrelevant for forest fire monitoring. Indeed, the objective of these sensors is not to measure the 4 different states of the system but only the transient state of the fire. The only possible change of state is from *tree* to *burnt tree* through the state of *burning tree*. If the sensor detects fire, we know that all previous states were *tree* and that all subsequent states will be *burnt tree*. We can imagine that these sensors are similar to smoke detectors, and as soon as the sensor detects fire, it sends a message to the state estimator which takes into account the measurement. Thus, even if the sensor does not survive the fire, this will not impact the state estimation because once the *burning tree* state is measured all the following states are identical. This type of sensor can also be cost-effective compared to the use of satellite imagery or heat-resistant mobile sensors.



(a) Sensor Map (b) Observability/Reconstructiblity Map

Figure 5.9. Map of the sensor network and the non-observable and non-reconstructible configurations. The non-observable configurations are shown in red and the non-reconstructible configurations in blue. Note that a non-reconstructible configuration is necessarily non-observable (see section 2.3.3).

By evaluating the observability and reconstructibility with algorithm 1 for the sensor network and the time horizon $T = 30$, we realise that out of the 5055 initial

configurations 527 are non-observable and 117 are non-reconstructible. By representing the configurations by the position of the fire at the initialisation, we can represent on the lattice the position of the non-observable and non-reconstructible cases (see Figure 5.9b). Firstly, we immediately notice that all non-reconstructible configurations are located in small areas of the forest that do not have any sensors. This means that the large area of 4917 cells is reconstructible for a time horizon $T = 30$. Then, the large concentration of non-observable configurations (especially in the right bottom corner) is justified by a low concentration of sensors in this area. A more efficient coverage of the sensors could have solved this problem.

The observability and reconstructibility evaluation for the 5055 configurations takes only a few dozens of seconds[1]. Without any simplification on the initial conditions, the number of initial configurations to evaluate amounts to 4^{10000}. Even knowing the topology of the forest and evaluating all possible cases (*i.e.* each non-empty forest elements can be either a tree, a burning tree, or a burnt tree), the number of configurations is far too high (3^{5054}) to be considered. For systems of this dimension, the use of simplification in the number of initial configurations is necessary to evaluate observability due to the exponential complexity.

Now that we have evaluated the reconstructibility, we can implement the state estimator as presented in section 3.3.3. Hence, the initial state of the state estimator $\hat{x}_0$ will be composed of the 5055 initial configurations used for the observability and reconstructibility evaluation. In the case where a configuration is not reconstructible, then there will be no convergence to a single current configuration. However, as all non-reconstructible configurations are non-reconstructible because there are no sensors in those areas, this means that they will be mistaken for an absence of fire. As the isolated areas are small, the total error will be small. Then, to be able to compare with synchronisation later on, we will study the evolution of the error as a function of time for this estimator. The estimated state of each cell will be calculated from the majority of the state of that cell over all current configurations present in $\hat{x}_t$. For example, if for a given cell c_i, 40% of the configurations in $\hat{x}_t$ have the *tree* state for that cell ($s_t(c_i) = 1$) and 50% have the *burnt tree* state ($s_t(c_i) = 3$) and the rest have the *burning tree* state ($s_t(c_i) = 2$), the estimated state of that cell will be **burnt tree** as it is the majority. In Figure 5.10, we have plotted the average error of the state estimator as a function of time. The error increases until $t = 5$ and then starts to decrease until it reaches a fixed value at $t = 29$. The starting time concerns the propagation of the fire before the sensors detect the fire, which is when the number of possible configurations is greatly reduced. The value at $t = 29$ is not zero (but $\epsilon_{29} = 2.9e^{-5}$) because of the 2.3% of non-reconstructible configurations.

To place the sensors correctly, it is not necessary to study the whole forest, but the individual areas. Indeed, as fire cannot spread from one area to another, each area

[1] The simulations were performed on a personal computer using Matlab without an y particular optimisation of algorithm 1.

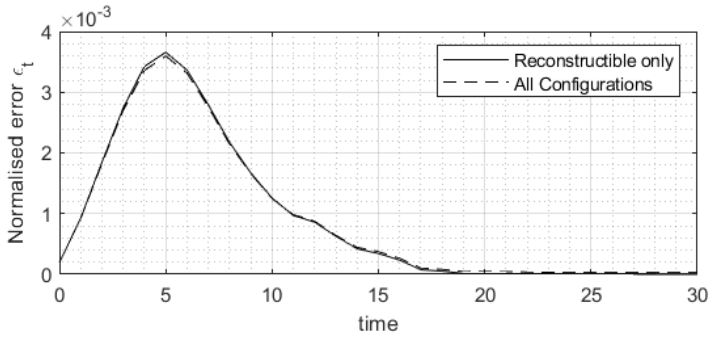


Figure 5.10. Evolution of the normalised synchronisation error as a function of the time for the state estimation of the sensor network presented in Figure 5.9a. The mean is computed through 10^3 simulations.

can be considered as independent. Thus, the observability and reconstructibility of each zone can be ensured independently and, in the same way as for decentralised observability, the observability of the system is the combination of the observability of each zone. Therefore, at least one sensor per zone is required. However, one question arises as to the interest of such a method. Indeed, the objective is to monitor the propagation of the fire, but if the fire breaks out in a very small area, there will be no propagation over the entire forest. Therefore, the minimal area to be measured depends directly on the objective of the monitoring. Should any fire be detected in the forest? Should the spread of the fire be measured in order to fight it?

At first glance, the placement of the sensors seems relatively simple. Indeed, by using randomly placed fixed sensors, we are able to ensure reconstructibility for a large majority of the lattice, 97.7% of the initial configuration are reconstructible. This is largely explained by the property of this class 2 CA for which three quarters of the states in $\mathcal{S}$ are absorbing states. If in our case, we wanted to ensure observability then we would have to spend more time on sensor placement. In that case, placing the sensors randomly or by trial and error would appear to be inefficient. We could use a heuristic sensor placement methods such as the Monte Carlo algorithm (Chakraborty *et al.*, 2020; Castello *et al.*, 2010), or evolutionary methods such as the genetic algorithm (Liu *et al.*, 2008) where the objective is to minimise the number of unobservable or unreconstructible configurations. The management of mobile sensor networks is much more complex than for fixed sensors, because in addition to the initial position, the sensor trajectories must be carefully managed. Demetriou and Hussein are working on coordination methods for mobile sensors in the context of observation (Demetriou and Hussein, 2009; Demetriou, 2010). Decentralised methods of coordination could also be used, such as the multi-agent systems used by Schlotterbeck *et al.* (2016) for forest fire monitoring. In the latter, the observation method used is more related to synchronisation than to observability, but the observability criteria could be included in the behaviour of the agents.

In conclusion, in this section we have been able to study the observability and reconstructibility for the trivial fire propagation model. The reduction in the number of plausible initial configurations makes it possible to calculate observability and reconstructibility for a CA with several thousand cells. In this example, we did not try to place the sensors in such a way as to solve the problem of monitoring the spread of a forest fire. However, the simplicity of the model makes it possible for reconstructibility with a random placement of static sensors. In the next section, we will study synchronisation. The simulations will be done with the same number of sensors and on the same forest topology in order to be able to compare the performances of two observation methods.

5.2.3 *Synchronisation*

The objective of this section is to use the results of chapter 4 on CA synchronisation for monitoring the propagation of a forest fire. For this purpose, we shall use the same hypotheses as for the previous observability analysis, *i.e.* the forest topology is known and the fire can only start in one of the cells. We shall be able to compare the performances of the usual synchronisation and the optimised synchronisation for low initial errors. We will start by augmenting the trivial fire propagation model presented in Figure 5.3 to better fit the use of synchronisation as a state estimator.

Contrary to what we did in chapter 4, the replica and the driver will not be initialised randomly but with the forest topology presented in Figure 5.8. This will allow us to compare more easily the results of this section with those from the previous section. Thus, only the position of the fire outbreak will be randomly initialised (among the 5054 possible positions). For each simulation, we will use a sensor network consisting of 150 randomly initialised sensors. By default we will use random synchronisation but we will present the performances of different types of sensors thereafter. The results on the synchronisation will be studied statistically through the average of 10^4 simulations. In addition, as the empty cells state values cannot change, the normalised synchronisation error will be calculated with respect to the number of cells whose state can effectively change. Therefore, in the calculation of the normalised synchronisation error (4.4), N will be taken as $N = 5054$ instead of $N = 10000$.

The fire spread model shown in Figure 5.3 is a simple way to represent the spread of a forest fire. We have also seen that it allows to cope with different parameters by playing on the density of the forest and the shape of the neighbourhood. However, this model is not well adapted for synchronisation purpose because its absorbing states do not propagate. Indeed, if a coupled cell measures a cell with the *burnt tree* state, then this cell state will be copied to the same cell in the replica. As a burnt tree

is measured, this means that fire was present on this cell not so long ago and that its neighbours are either burning or burnt. It is not possible for a burnt tree to be directly adjacent to a tree because we have not implemented fire extinguishment and there are no burnt tree cells in the initial configuration. However, with the current model, the burnt tree that was just measured will not impact the neighbouring cells of the replica even though we know that the neighbouring cells are necessarily burnt or burning. Therefore, we will augment the replica model so that a tree cell next to a burnt tree cell becomes a burnt tree cell, as shown in Figure 5.11.

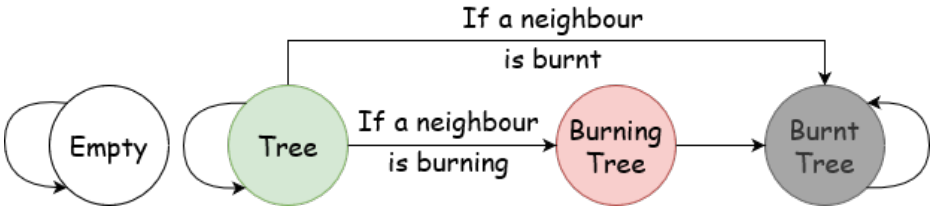


Figure 5.11. Diagram of the state changes of augmented trivial fire propagation model.

In Figure 5.12 we show the evolution of the normalised synchronisation error for both models. For each simulation, the sensors were positioned identically for both models. The normalised synchronisation error of the augmented model is significantly lower (*i.e.* 66% lower) than the error for the classical model. The error starts to decrease earlier in the augmented case (at $t = 10$, compared to $t = 17$ for the classical model) due to the propagation of the ash, so that the replica can start to synchronise as soon as burnt trees are measured. On the other hand, we notice that perfect synchronisation is not reached in the time horizon $T = 100$ for any of the two models. The error for the augmented model is really low compared to the classical model ($\epsilon_{100} = 3.9e^{-5}$ for the augmented model and $\epsilon_{100} = 6.0e^{-4}$ for the classical one). It is not until $t = 150$ that the classical model reaches such a low error. Like in the previous section, it is necessary that all isolated areas are measured. In that sense, the augmented model does not provide a more efficient way to measure these isolated parts of the CA. However, measuring only one burnt or burning cell from these isolated parts is enough.

As we have seen in chapter 4 with Figure 4.5 and Figure 4.6, the type of sensor used for the coupling of the cells is of a great influence on the synchronisation performance. This difference also depends largely on the system we are studying. For this purpose, we will study in the case of the forest fire spread for 3 types of sensors: fixed sensors, mobile sensors and random sensors. We had already noted that the random sensors were mobile sensors with a speed largely higher than the dynamics of the system, therefore, we will study mobile sensors with relatively low speeds. We will study two sensor networks with mobile sensors. At each iteration, the sensors will move to a

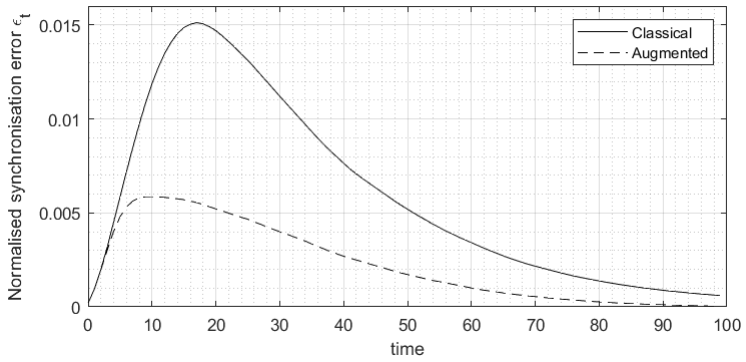
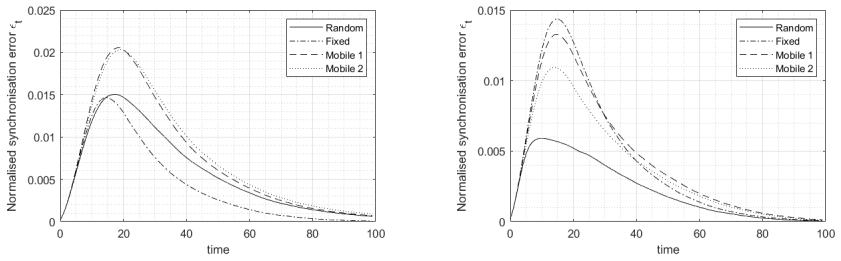


FIGURE 5.12. Evolution of the normalised synchronisation error as a function of the time for the classical and augmented fire propagation model. The mean is computed through 10^4 simulations.

valid cell (*i.e.* not an empty cell) randomly chosen in its range of motion. For this study, we will use two mobile sensor networks, one with a speed of 1 and the other with a speed of 2. A speed of 1 corresponds to the speed of propagation of fire. To compare the performance, the four sensor networks will be initialised in the same way, the 150 sensors will have a random position which will be the same for the four sensor networks. Figure 5.13 shows the average synchronisation error for these 4 sensor networks with the classical model and the augmented model for the replica.



(a) Classical Model (b) Augmented Model

FIGURE 5.13. Evolution of the normalised synchronisation error as a function of the time for the classical and augmented fire propagation model. The mean is computed through 10^4 simulations.

For the classical model (see Figure 5.13a), the fixed sensor network performs best, both with respect to the time at which the error starts to decrease and to the error value at $t = 100$. The other three sensor networks have approximately the same time for the error to decrease and the same final value. The difference between the mobile and the random sensors is the growth of the synchronisation error at the beginning which is 25% lower for the random sensors. With a high speed of movement, the random sensor network can scan the whole lattice and measure more cells, even if these cells are mostly trees and burnt trees. The performance of the fixed

sensor network is explained by the non-displacement of the sensors. Indeed, when the front progresses on the lattice, the mobile sensors can skip a cell and measure ashes before measuring fire, this cannot happen with fixed sensors. They will never measure ash before measuring fire. This also explains why the performance of the fixed sensor array is identical regardless of the model used. With the augmented model (see Figure 5.13b), the mobile and random sensors benefit from the model augmentation because they can measure ashes before fire. In addition, the faster they move, the further they can measure burnt trees from their initial position. Comparing the two mobile sensor networks, it can be seen that increasing the speed increases the synchronisation performance. Furthermore, as the random sensors can be considered as mobile sensors with a very high speed, it can be easily conjectured that the random sensor network is the limit of the mobile sensor network when increasing the speed. Thus, without adding any additional behaviour, the combination of augmented model and random sensor network seems to be the most suitable to solve this problem. It is also important to note that if we want to use the optimised synchronisation, it works when the sensors are moving fast. If they are too slow, they will not be able to reach the error area before the fire has spread on a large part of the lattice, which makes the optimised synchronisation useless.

In section 4.4.3, we presented a method of coordinating the sensor network when the initial error is small. We were able to apply the results to the fire propagation model by adding a dimension to the lattice. Therefore, the error propagation no longer has a triangular shape but a pyramidal shape with a square base (because this is the shape of the neighbourhood). The height of the pyramid represents the time. At each iteration, the error can propagate to 8 other cells (see Figure 2.1b for the shape of the neighbourhood), but to only 1 in each direction (including the diagonals). By making this slight change, we are able to apply the optimised synchronisation method to the forest fire propagation model. In Figure 5.14 we have plotted the average evolution of the normalised synchronisation error for the classical model and increased using the classical and optimised synchronisation (presented in section 4.4.3). Figure 5.15 shows the same error but on a semi-log scale for a more detailed study of the evolution of the error.

Firstly, we notice that the time at which the error starts to reduce is reduced, the type of model used seems to have no impact when used with the optimised synchronisation. The time is $t = 5$ for the synchronisation and $t = 10$ and $t = 17$ for the classical synchronisation with respectively the augmented and classical model. This decreasing time is the same as for the use of state estimation with reconstructibility (see Figure 5.10), but we will compare the two methods in more detail in the next section. Then, we notice that the decrease of the synchronisation error is stronger with the use of the optimised synchronisation (which is even more important when paired with the augmented model), this can be seen very well in Figure

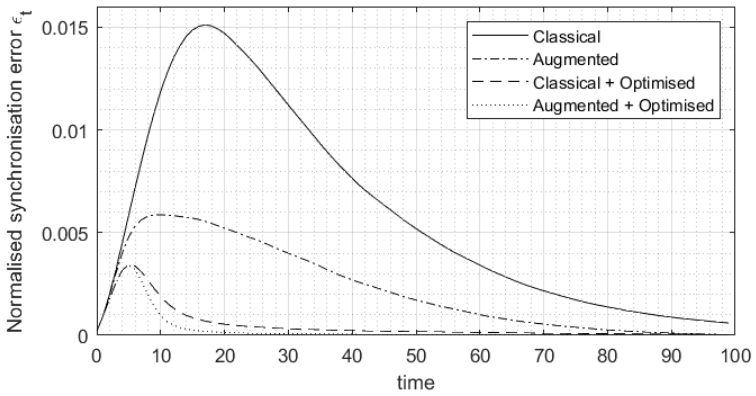


FIGURE 5.14. Evolution of the normalised synchronisation error as a function of the time for the classical and augmented fire propagation model and the classical or optimised synchronisation. The mean is computed through 10^4 simulations.

5.15, where the decrease is linear for the classical synchronisation and exponential for the optimised one. However, the impact of the optimised synchronisation is lower when the time increases. At $t = 20$ the decrease becomes linear similar to the one using the classical synchronisation. The use of the augmented model results in a 17% faster error decrease, which explains why the error of the augmented model with the classical synchronisation catches up with the classical model with the optimised synchronisation (at $t = 97$). On the other hand, none of the four configurations provide a null error at $t = 100$. Finally, it can be concluded that the use of the augmented model with optimised synchronisation seems to be the most suitable for monitoring, as the rapid reduction of the synchronisation error provides a faster information on the fire propagation.

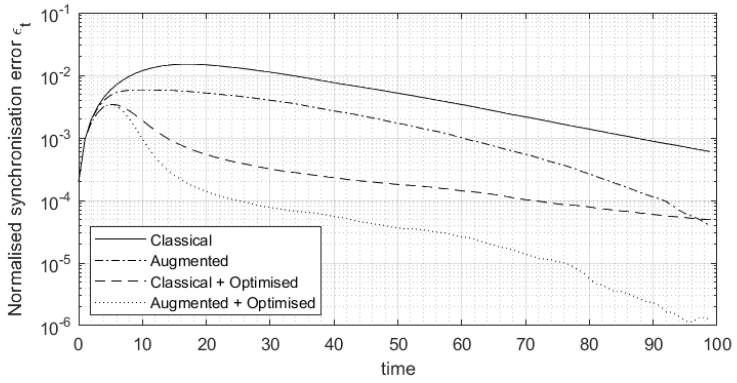


FIGURE 5.15. Evolution of the normalised synchronisation error on a semi-log scale for the classical and augmented fire propagation model and the classical or optimised synchronisation. The mean is computed through 10^4 simulations.

As in the case of rule **R30** (see Figure 4.20b), the synchronisation error with the augmented model and the optimised synchronisation at time $t = 100$ is not perfect because there are a number of simulations (1.9% at $t = 50$, 0.63% at $t = 75$ and 0.17% at $t = 100$) where the error is non-zero. In 70% of the cases, these are small errors that group fewer than 10 cells. To compensate for this, the coordination should be improved and the areas should be measured in a regular and balanced way. This will ensure that the fire front is detected sooner and the earlier it is detected, the more effective the optimised synchronisation is due to the error zone size increasing with time. To enable good sensor management, Schlotterbeck *et al.* (2016) use a set of forces to ensure that the sensors are evenly distributed over the space. This method could be adapted to this model by making the empty areas have a repulsive force as borders can have. Other methods such as fair space division have been studied in (Plénet *et al.*, 2020b) allowing sensors to move randomly but only within a given area. In this way, the sensors share the space, but can also concentrate when the error is detected so that optimised synchronisation can be applied.

In this section, we have been able to develop an efficient state estimator using synchronisation. To do this, we had to augment the replica model to take into account a model-specific phenomenon. We also studied the types of sensors and concluded that the use of the augmented model with random sensors was the best performing pair. Finally, we implemented the optimised synchronisation presented in chapter 4 for the monitoring of forest fire spread. In the next section, the last one concerning the forest fire study, we will compare the results of the state estimator presented in this section with the one presented in the previous section.

5.2.4 Conclusion

In the previous two sections, we have developed two state estimators. The first one is based on reconstructibility while the second one uses synchronisation. In Figure 5.16, we have represented the average evolution of the errors in both cases. It is immediately noticeable that the performances are similar, the synchronisation admits a faster decrease of the error between $t = 5$ and $t = 15$, but has a slightly higher error at $t = 30$. In the case of synchronisation, this error will continue to decrease as more measurements are taken by the sensors. On the contrary, the error in the case of state estimation with reconstructibility will be stable. In both cases, these errors represent a small portion of the simulations that have low error rates. However in the case of synchronisation this error may be over the large, connected, forest area.

Although the average performance of both estimator is very similar, the two state estimators are very different. The first one, based on reconstructibility, uses a fixed sensor network, so it could be improved to make the sensors move and measure

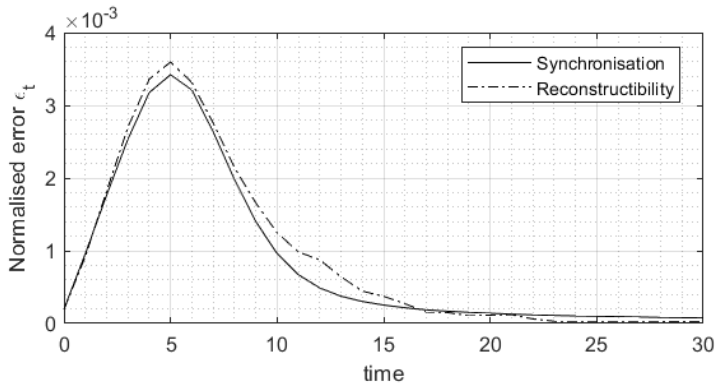


Figure 5.16. Evolution of the normalised synchronisation error for the augmented/optimised synchronisation of Figure 5.14 and the state estimation of Figure 5.10.

isolated areas. However, the state estimator algorithm is much more algorithmically heavy as it requires the simulation of several thousand configurations simultaneously. On the other hand, the synchronisation uses a network of mobile sensors with the presence of coordination (optimised synchronisation). The choice of one method or another will depend on the types of sensors available and the computing power available for the state estimator. On one hand, the characterisation of reconstructibility is more interesting as it ensures that the state will be reconstructed correctly. However, as long as the state estimator is not error tolerant this method cannot be applied in consideration of real scenarios. In addition, reconstructibility must be assessed for every small variation in the system, including wind, forest topology, etc. On the other hand, synchronisation, provides more freedom in coordination as reconstructibility does not need to be verified. It offers an easier framework to use decentralised coordination method such as multi-agent systems or wireless sensor network. The study of synchronisation through statistics must be done with care to avoid scenarios where the error does not synchronise at all.

With this example, we were able to present a method for setting up a state estimator. We started by making a trivial model of fire propagation. Then, we were able to study its observability and reconstructibility as well as the synchronisation. We were able to compare the two estimators in terms of performance and the advantages and disadvantages of each. In the next two examples, we will address notions not used in this example, such as the decentralisation of the observation or the observability of the ACA.

5.3 Monitoring of a Road Network Through a Toll Booth

In this example we will study the observability and reconstructibility of a road traffic model. We plan to use fixed sensors as these are already present on the road as radar or road tolls. Several road traffic models have been developed since the 1970s, the most famous being the ECA **R184**. However, the lack of granularity in traffic modelling has led to the use of other types of models.

Nagel and Schreckenberg (1992) propose a specific traffic model for motorways by adding the notion of speed. With this model, they manage to model typical traffic behaviours such as traffic synchronisation or wide moving jams. Several years later, Knospe *et al.* (2002) extended this model to a two-lane road. Other models are proposed, especially for city traffic where the modelling is based on the presence of traffic lights. With a CA model, Brockfeld *et al.* (2001) propose to optimise traffic lights to maximise traffic in cities.

Usually, a road has unidirectional dynamics in the direction of traffic, in the case where there are several lanes, each of these have little interaction with one another. We could therefore conveniently apply observability or reconstructibility because of the few boundary cells. However, the complexity of this method requires the use of a small number of cells and states, which is why we will study the simplest traffic model, the ECA **R184**.

5.3.1 Presentation of Wolfram Elementary Rule R184

R184 is one of the most used of Wolfram's elementary rules. It has been used in many fields to model physical systems such as traffic flows (Maerivoet and De Moor, 2005) and the particle deposition (Krug and Spohn, 1988). As with all ECA, **R184** is a boolean, one-dimensional, deterministic CA but there are a number of variants of this rule. Rosenblueth and Gershenson (2011) augmented the model to simulate traffic light intersections. The cells of the intersection change their rule as a function of time to allow vehicles to pass or not. This method will be presented in detail in section 5.3.3 when we study the road network of a city. Higashi *et al.* (2021) have transposed the **R184** rule to Fuzzy Cellular Automata, which allows them to have a car density rather than a boolean value. Nishinari and Takahashi (1998) showed that **R184** is a special case of the discretisation of the Burgers equation, a partial differential equation used for modelling gas dynamics, acoustics or road traffic. But finally, we will focus on the simple case of the ECA **R184** for the moment.

In **R184** as a traffic model, the state of the cells represents the presence, or not, of a car as shown in Figure 5.17. All cars move in the same direction, in this case to the right, and they can only move forward if they have an empty space in front of

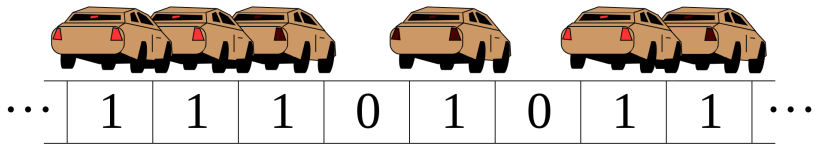
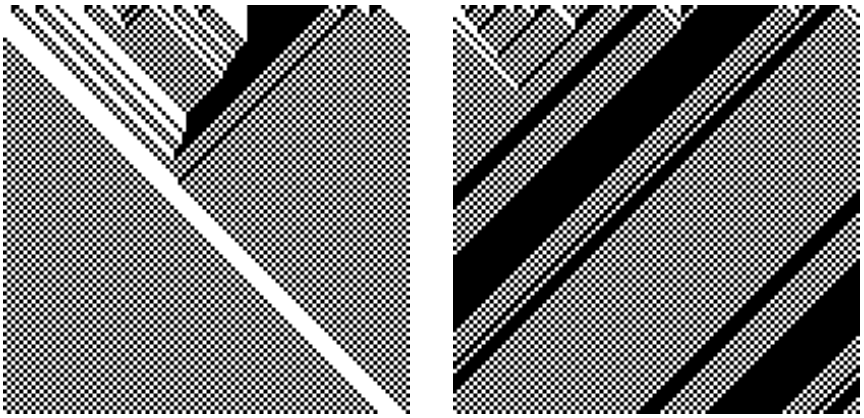


FIGURE 5.17. Wolfram elementary rule **R184** interpreted as a simulation of traffic flow. Each 1 cell corresponds to a vehicle, and each vehicle moves forward only if it has open space in front of it. Image from David Epstein, English Wikipedia.

them. Therefore, to have a continuous flow of cars, you need a maximum of $N/2$ cars for a lattice of N cells otherwise traffic jams will appear. On Figure 5.18, we present two simulations of the CA **R184-100-PB**, in the first case, the traffic jam (in black) disappears because the road is not saturated. In the second case, on the contrary, the traffic jam only moves in the opposite direction of the cars.



(a) **R184-100-PB** with 46 cars (b) **R184-100-PB** with 64 cars

FIGURE 5.18. Example of the evolution of **R184-100-PB** for different numbers of cars on the network. The dense black parts represent traffic jams. Time is going down on the vertical axis.

On the opposite to what we were able to do in the previous example, we cannot make simplifications on the initial configurations of the system. The number and position of vehicles is unknown at the initialisation. Because of this, we cannot study a CA with more than a dozen cells. Therefore, we will study the ECA **R184-10-PB**. The periodic boundary ensures that there are always cars in the system, the cars leaving the system are also entering it at the other end. Thus, using definition 2.1, the CA can be defined as :

- $\mathcal{L} = [\![0; 9]\!]$
- $\mathcal{S} = \{0, 1\}$

- $\mathcal{N}(c_i) = \{c_{i-1}, c_i, c_{i+1}\}$, with the periodic boundaries conditions $c_{-1} = c_9$ and $c_{10} = c_0$.
- $f\colon s_t(\mathcal{N}(c_i)) \mapsto s_{t+1}(c_i) = s_t(c_{i-1}) \oplus s_t(c_i) \cdot [s_t(c_{i+1}) \oplus s_t(c_{i-1})]$, with respect to the modular arithmetic of $\mathcal{S}$ as presented in eq. (2.6).

Now that we have defined CA, we can study its observability and reconstructibility. We will also use decentralised reconstructibility in order to increase the size of the road we can monitor. Then, in the last section, we will extend this model to have a 2-dimensional road network with intersections in order to study its reconstructibility.

5.3.2 Study of Observability and Reconstructibility

We will study the observability and reconstructibility for the ECA **R184-100-PB** presented in the previous section. As this CA is non-linear, we will use the results from section 3.3. However, as we are studying all initial configurations and it is not very large, we will use the matrix approach presented in section 3.3.1. The matrix L of the transition function F is of size 1024×1024 and this is quite a reasonable size for doing multiplication operations. We did not construct the matrix L with the methods using the semi-tensor product, but as an adjacency matrix, *i.e.* by evaluating which configuration s_{t+1} corresponds to $F(s_t)$.

This CA poses another problem regarding observability: it cannot be evaluated with few sensors. Indeed, to differentiate the two initial configurations presented in Figure 5.19, it would be necessary to observe 2 of the 3 cells that are non-zero among the two initial configurations. This problem can be shifted over the whole CA and it would thus be necessary to place a sensor every two cells to resolve this issue. Conversely, this convergence of two initial configurations does not pose any difficulty with respect to the reconstructibility. Because of this, we will not try to find a sensor network that ensures the observability but only the reconstructibility.

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

(a) Configuration 0010100000

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

(b) Configuration 0001100000

Figure 5.19. Evolution of **R184-10-PB** for two similar initial configurations. Time is going down on the vertical axis.

We have the intuition that reconstructibility can be provided by a single sensor, wherever it is. Indeed, because the system propagates in one direction only, a sensor could see all the non-zero states pass and deduce their future positions. To carry on with the traffic flow analogy, this sensor could be a tollbooth which, at a certain moment, measures all the cars. However, as our objective is to study a large road network through decentralisation, we will place, as shown in Figure 3.5, a sensor at each end of the lattice. Thus, by measuring the c_0 and c_9 cells with fixed sensors, the sensor network can be defined as :

- $\mathcal{L}_q = \{0, 9\}$
- $H\colon \theta_t := H(s_t) = \begin{bmatrix} s_t(c_0) \\ s_t(c_9) \end{bmatrix}$

To verify the reconstructibility of the CA, we will first construct the observability matrix O_T from the matrix L and the matrix of H. Then observability and reconstructibility are respectively verified with theorems 3.5 and 3.6. As we cannot apply the Cayley-Hamilton theorem, we start by assessing the observability and reconstructibility with a minimum time horizon $T = 0$ and will increase it gradually until one of them is verified. If neither the observability nor the reconstructibility is verified by $T = 20$ we will stop here and look for another sensor network. In addition, to the evaluation of observability and reconstructibility, we count the number of unobservable and unreconstructible configurations and plot this number as a function of the time horizon T in Figure 5.20. After a few seconds of simulation, we find that the system is reconstructible (but not observable as we expected) from $T = 13$. Increasing the observation horizon does not reduce the number of unobservable configurations beyond 75% of the total number of configurations.

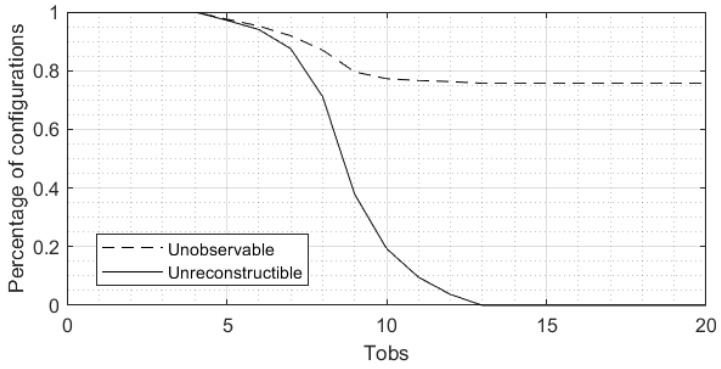


FIGURE 5.20. Percentage of unobservable/unreconstructible configurations as a function of the time horizon T for the ECA **R184-10-PB**.

Finally, we aim to extend our model to observe a larger road network. For this, we intend to observe a CA of 1, 000 cells but the algorithmic complexity of algorithm 1 is far too important. Instead, we plan to divide the CA of 1, 000 cells into several sub-observers of equal, smaller sizes. However, the complexity of the decentralised reconstructibility presented in section 3.4.2 is far too important to for a sub-observer of size 10. Indeed, with a time horizon $T = 13$, there will be 4^{13} possible boundary trajectories. We must use a sub-observer of smaller size in order to decrease the reconstructibility time horizon. Therefore, we will use 200 sub-observers with 5 cells, each of them measured by the sensor topology shown in Figure 3.5, one sensor at each end of the lattice. Similarly to what we have done for the study of reconstructibility, we have no constraint on the time horizon T, except that it must be larger than 5, the time needed to ensure reconstructibility if we consider the previous system with only 5 cells. Figure 5.21 shows the evolution of the percentage of boundary trajectories that are not reconstructible, *i.e.* boundary trajectories for which the CA is unreconstructible.

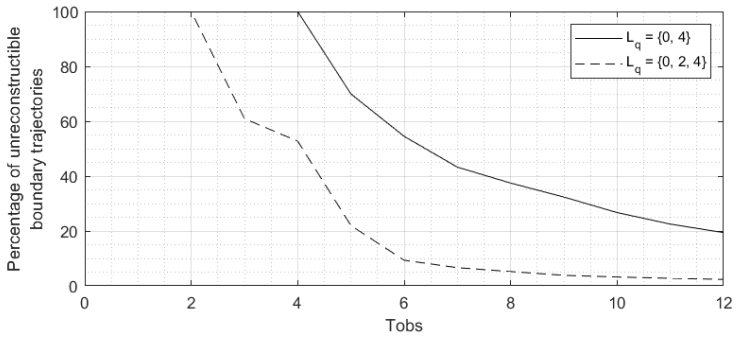


Figure 5.21. Percentage of unreconstructible boundary trajectories as a function of the time horizon T for the ECA **R184-5-UB**.

The sub-observer is never reconstructible for a time horizon shorter than 12. Beyond that, the time to check reconstructibility is too long for our machine. As an attempt to still ensure reconstructibility, we have added a sensor in the middle, the c_2 cell, of the sensor network. Even though the number of unreconstructible trajectories is lower, the system is still not reconstructible. The reason for this is that if the right-end boundary cell is in the 1 state for the entire duration, then the cars in the CA cannot move out of the lattice. It is therefore impossible to measure all of them and we cannot distinguish between the 00001 and 00011 configurations. And this is true for a time horizon as large as it is. To achieve reconstructibility, we would need to be able to detect when a large traffic jam is forming and optimise the placement of the sensors to ensure that we measure the blocked cars.

As a small part of the configurations are unreconstructible, we could do as in

the previous example and perform a state estimator and consider that it will work. However the unreconstructible configurations represent the whole lattice, not just small isolated areas as in the fire propagation model. The state of the sub-observer (except for the measured areas) will be unknown. With the use of measured boundaries, this error will not propagate and can be identified by the state estimator if the state does not converge. If the boundaries were not measured, the error would propagate throughout the system and affect the proper functioning of the observation. In this case, it would be necessary to think about implementing error management.

This method with the sub-observers allowed to ensure the reconstructibility for the whole CA which would have been impossible without it. Indeed, with 1000 cells the number of operations to evaluate the reconstructibility is 2^{1000} with the usual method against $2^{10} \cdot 2^{2*7} = 2^{24}$ for the reconstructibility with the sub-observers.

5.3.3 *Extending the Model to New York Traffic*

The model studied in the previous section represents a straight road without any particular signalling or slowing down (other than that caused by overloading the network). Rosenblueth and Gershenson (2011) have added the possibility of having intersections governed by traffic lights. Their model uses only elementary rules, **R184** for traffic roads and **R252** and **R136** for traffic lights. As the traffic lights change with time, so do the cell rules (see Figure 5.22). The central cell does not change its rule but its neighbourhood depends on the time, it always contains the two cells that are not restricted by the lights. As the transition function will change over time, the evaluation of observability and reconstructibility is time-dependent. Indeed, the O_T matrix will be different depending on the starting time, which was not the case with a time-invariant transition function.

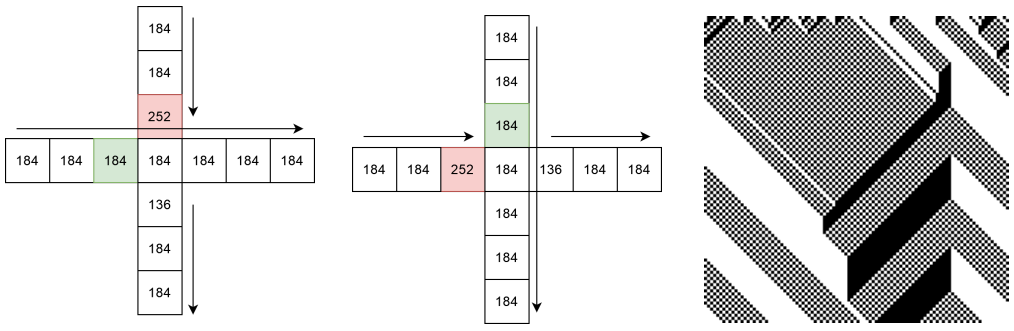


FIGURE 5.22. Elementary Rule of the cells according to the traffic lights.

For this example, we will use the lattice shown in Figure 5.22, *i.e.* with 13 cells.

We will use periodic boundaries, the vertical and horizontal lines will be looped on themselves. As far as the sensors are concerned, we will use 5 of them: one on each end and one in the centre, on the intersection.

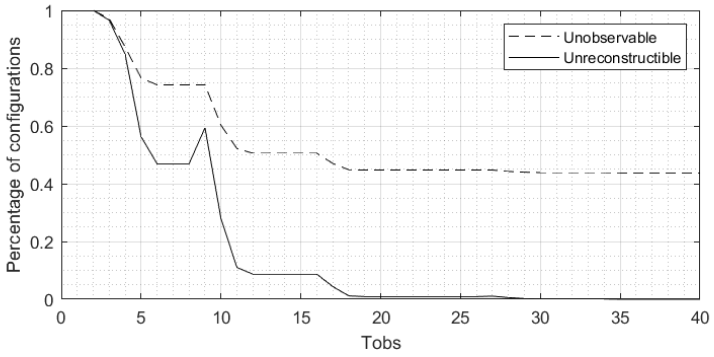


Figure 5.23. Percentage of unobservable/unreconstructible configurations as a function of the time horizon T for the intersection CA.

In the same way as before, we will evaluate the observability and reconstructibility using the method presented in section 3.3.1. Due to the large number of configurations, we will use the sparse matrix representation for the L matrix. In Figure 5.23, we show the evolution of the number of unobservable/unreconstructible configurations as a function of the time horizon. We see that the system is unreconstructible from $T = 30$ but not observable. Given the time needed to ensure reconstructibility, we will not study decentralised reconstructibility.

If we ever manage to ensure decentralised reconstructibility for both the intersection and the road (see previous section), then we would be able to build a square road network, similar to the one in Manhattan. Indeed, observability and decentralised reconstructibility work as long as the values of the boundaries are provided to them, no matter which CA is next door. This means that a modular state estimator can be constructed from elementary blocks. The time horizon for the reconstructibility of the system will be the longest horizon of all sub-observers.

5.3.4 Conclusion

In this example, we have studied **R184** and in particular its interpretation as a road traffic model. We were able to study its observability and reconstructibility with the theorems 3.5 and 3.6 because there were no constraints on the initial conditions. The observability could not be verified because it requires too many sensors. Once the reconstructibility was verified, we wanted to verify the decentralised reconstructibility in order to increase the size of the studied route. For this purpose we want to divide

the lattice into sub-observers of 5 cells. However, the reconstructibility could not be verified for the chosen time horizon and can never be verified with the chosen sensor network. The use of toll booths (*i.e.* static sensors at the beginning and end of the road) cannot ensure reconstructibility. A different sensor network with one or more sensors, mobile or not, between the sensors at the ends must be used. The purpose of these additional sensors would be to measure the state of the traffic jam when the tolls are no longer effective. In addition, we could put conditions on the values taken by the borders so that they are more faithful to reality. A traffic jam cannot last forever.

Subsequently, we chose to study an extension of the model proposed by Rosenblueth and Gershenson (2011) which allows to model an intersection managed by road traffic. We were able to verify the reconstructibility of this CA using 5 sensors. This intersection model allowed us to demonstrate the use of reconstructibility for a CA whose dynamics depend on time. In the end, if we had been able to verify the reconstructibility of the two models studied, we would have 2 sub-observers that would have allowed us to build a modular state estimator for a square road network similar to that of New York.

5.4 Random Number Reconstruction

In cybersecurity, pseudo-random number generator (RNG) are a crucial element in many applications (Marton *et al.*, 2010). They are used to secure *https* connections with *SSL*, to secure connections to wifi networks with *WPA2* and many others. The robustness of RNG depends on both the probability distribution and the predictability of the generator. It is evaluated using randomness tests which allow the quality of the RNG to be assessed according to several criteria. Among these tests, the best known are the Diehard tests (Marsaglia, 1996) and the TestU01 (L'ecuyer and Simard, 2007). As a crucial element of cybersecurity, RNG are a source of several vulnerabilities that are grouped under the term "Random Number Generator Attack" (Kelsey *et al.*, 1998) where the objective is to obtain information about the past or future generations of random numbers.

Cellular automata random number generators (CA RNG) have been widely studied for more than three decades (Chaudhuri *et al.*, 1997). The local, simple and regular interactions of CA make them a good way integrated at large scales. Initially, CA RNG are one-dimensional (Chaudhuri *et al.*, 1997; Hortensius *et al.*, 1989; Tsalides *et al.*, 1991) but CA with two or more dimensions turn out to generate random numbers of better quality Chaudhuri *et al.* (1997); Tomassini *et al.* (2000) but are more complex to set up. For that, different algorithms have been developed to build "automatically" these CA RNG (Tomassini *et al.*, 2000).

In this example, we will use the observability of affine cellular automata (ACA) to

reconstruct present, past and future random numbers using information obtained by a random number generator attack. The objective of the attack is to obtain partial information on the state of the CA RNG that will subsequently be used to apply the results of section 3.2. The nature of the attack, as well as the method used, does not matter in the context of applying the Kalman criterion, only the information obtained from the attack is important. The attack will be seen as a sensor that observes the state of the random number generator and therefore represented by the output operator H which will depend solely on the information obtained by the attack. Of these three examples, the first will use knowledge of a single random number to deduce past and future numbers, the second will use information about a single bit of the random number and the last will be based on the binary parity of the generated random number. Throughout these examples, we will use the Tomassini *et al.* (2000) 2D CA RNG which we define in the next section.

5.4.1 Cellular Automata Random Number Generator

For this example, we will study a CA RNG generating high quality random numbers proposed by Tomassini *et al.* (2000). The quality of the random numbers generated by this CA RNG was evaluated according to the *Diehard* tests defined by Marsaglia (1996) which were all passed successfully. This CA RNG is a two-dimensional Boolean CA that generates a number consisting of 64 hexadecimal random digits. The CA generates a random number every 4 iterations but for the simplicity of the example, we will consider that it generates a 64 bits random number every iteration rather than a 64 hexadecimal digits random number every 4 iterations.

The CA RNG is defined as follows:

- $\mathcal{L} = \{0, 1, \ldots, 7\} \times \{0, 1, \ldots, 7\}$
- $\mathcal{S} = \{0, 1\}$
- $\mathcal{N}: c_i, j \mapsto \{c_{i-1,j}, c_{i,j-1}, c_{i,j}, c_{i+1,j}, c_{i,j+1}\}$ with null boundaries so $s(c_{-1}) = 0$ and $s(c_8) = 0$.

In their paper, Tomassini *et al.* describe three boundary conditions: cyclic, fixed (full) and fixed (reduce). We can easily model fixed (full) and cyclic boundary conditions but the reduced version requires to model an CA of 10×10 but only the 8×8 cells in the middle represents the random number. In our example, we will use the fixed (full) version with a zero value at the boundaries.

The CA RNG of Tomassini *et al* is a hybrid CA, the local transition function f depends on the cell position. Figure 5.24 describes the transition function that

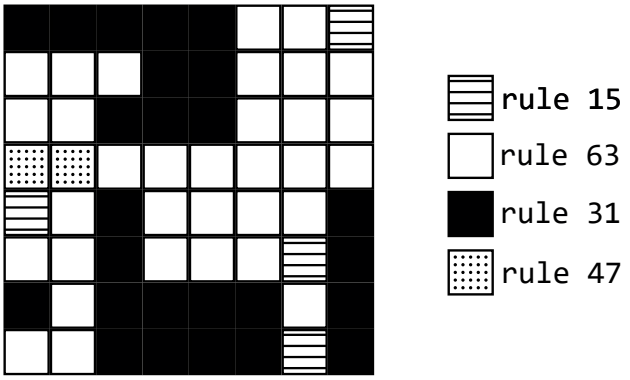


Figure 5.24. A 8×8 hybrid CA RNG proposed by Tomassini *et al.* (2000). The colour of the cells represents the transition function associated to this cell according to equation (5.3).

applies to each cell as a rule whose value is defined by the 6-bit string *XCNWSE* which corresponds to the following transition function:

$$\begin{aligned} s_{t+1}(c_{i,j}) = X + C \cdot s_t(c_{i,j}) + N \cdot s_t(c_{i-1,j}) + W \cdot s_t(c_{i,j-1}) \\ + S \cdot s_t(c_{i+1,j}) + E \cdot s_t(c_{i,j+1}) \end{aligned} \tag{5.3}$$

Therefore, according to the previous generic transition function, the rules 15, 31, 47 and 63 correspond respectively to the transition functions :

- $s_{t+1}(c_{i,j}) = s_t(c_{i-1,j}) + s_t(c_{i,j-1}) + s_t(c_{i+1,j}) + s_t(c_{i,j+1})$
- $s_{t+1}(c_{i,j}) = s_t(c_{i,j}) + s_t(c_{i-1,j}) + s_t(c_{i,j-1}) + s_t(c_{i+1,j}) + s_t(c_{i,j+1})$
- $s_{t+1}(c_{i,j}) = 1 + s_t(c_{i-1,j}) + s_t(c_{i,j-1}) + s_t(c_{i+1,j}) + s_t(c_{i,j+1})$
- $s_{t+1}(c_{i,j}) = 1 + s_t(c_{i,j}) + s_t(c_{i-1,j}) + s_t(c_{i,j-1}) + s_t(c_{i+1,j}) + s_t(c_{i,j+1})$

Regardless of the rules number, the transition function (5.3) is an affine map. Therefore every CA that respect Tomassini's definition can be represented with the state representation (2.12). However, as the CA RNG is hybrid we cannot construct the matrix A as a Toeplitz matrix (see section 3.2.3). The "shape" of the matrix remains the same but the value of the a_i change depending on the value of the rule (*e.g.* a_0 is the value of the bit C).

To construct these CA RNGs, Tomassinni et al. use a genetic algorithm with random tests as fitness function. Moreover, the presence of large cycles is also a guarantee of quality because it allows the generation of a very large number without "looping". Hence, they are able to easily generate CA RNG with different sizes that generate good quality random numbers (from the point of view of DieHard tests).

Additionally, the CA RNG presented in Figure 5.24 is reversible, which does not seem to be a property sought by the authors but rather a consequence of the criteria of the method used.

In the following sections, we will present two purely theoretical attacks which aim to show a use of Kalman criterion. In each of the examples, we will briefly present the attack and the associated output operator that reconstruct the state of the system. However, the details of the calculations will not be presented due to the large number of cells.

5.4.2 *Regular Measurement Attack*

In this first example, we study the observability of the CA RNG where the sensors have partial information on the generated random numbers. The aim is to study the observability when the sensor measures one bit of information on each iteration. We could have used more information bits, which would have potentially decreased the observability time horizon as a consequence of the Cayley-Hamilton Theorem. As specified in the introduction of this section, we are not interested in the "physical" method used to obtain the information on the CA RNG, for example this could have been recovered using an intrusive hardware attack (Samyde *et al.*, 2002) or side-channel attack (Standaert, 2010).

To reconstruct random number stream, we need only one bit of information on the random number at each time step. The measurement can be done on a single digit (read digit as the position of the bit in the random number) or can change with time. The only information needed is the position of the measured digit at each time step. Therefore, we can create the operator C_t which corresponds to the digit of the random number (or the cell associated with this bit) measured at time t. The operator C_t is of the form :

$$C_t = \begin{bmatrix} 0 & \dots & 0 & 1 & 0 & \dots & 0 \end{bmatrix}$$

, where the 1 corresponds to the measured bit.

Before assessing observability, we will check whether the output operator is adapted for observation. As we specified in the previous section, Tomassinni's CA RNG are constructed to be reversible and therefore injective. From definition 2.7, the injectivity of the CA guarantees that all output operators are suitable for observation. Moreover, remark 2.3 specifies an equivalence between observability and reconstructibility, so it will not be necessary to evaluate reconstructibility if observability is not verified.

Then, we need to know the operators C_t associated with the measurements y_t and from there construct the observability matrix O_T over a time horizon 64. The system

is observable (*i.e.* rank $O_T = 64$) regardless of the bit measured in the static case as well as for the different trajectories we have tried. However, it is not reasonable to test all the possible trajectories (there are 64^{64} trajectories) so we will still have to verify if the system is observable with the given trajectory. Using the corollary 3.1, it is possible to find the initial generated random number (*i.e.* the state x_0) and to deduce the future generated random number until the CA RNG is initialised with another seed (a pseudo-random generator process that consists of manually setting the state of the CA RNG).

Furthermore, since we know that the CA RNG is reversible, it is possible to deduce the previous states of the system. Indeed, the matrix A is invertible and given the initial state x_0 reconstructed using the Kalman criterion, we can express all states x_i as:

$$\begin{cases} x_t &= A^t \cdot x_0 + J_{t-1} \cdot \eta \\ x_{-t} &= A^{-t} \cdot x_0 - J_t^{-1} \cdot \eta \end{cases}$$

where A^{-t} is the inverse of A to the power t and $J_t^{-1} = \sum_{k=1}^{t} A^{-k}$

In other words, once we have successfully obtained a random number, we are able to deduce all future and past random numbers. In the next example, we deal with the case where the sensors cannot obtain a particular bit but a property of the whole number, the parity bit of the random number.

5.4.3 *Parity Attack*

The objective of this attack is to reconstruct the random number sequence using only the parity bit of the random numbers. The parity bit is the binary sum of all the bit of the number. With this method, it is possible to reconstruct the initial state even if the random number information is not directly accessible. One method that seems to be feasible is to correlate the energy consumption of the system with the memory consumption (Standaert, 2010).

Regardless of the method used, we assume to have access to the binary parity of the random number generated at each iteration. Therefore, the state of the system can be represented by the following:

$$\begin{cases} x_{t+1} &= Ax_t + \eta \\ y_t &= \sum_{i=0}^{64} x_t^i \end{cases}$$

The sum uses binary modular arithmetic.

The output operator C_t associated with parity is a row vector composed of 64 1 which does not depend on time. From there, the observability matrix can be computed with $T = 64$ and it is full rank matrix. The system is observable with the parity measurement, which allows us to reconstruct the initial state as well as the past and future states due to the reversibility of the CA RNG.

5.4.4 Conclusion

In those two examples, we have been able to reconstruct the random number stream from the partial information about the random numbers. The information obtained by making attacks on the random number generators is not described because the reconstruction of the state is independent of these methods. These attacks vary according to the hardware or digital support used by the CA RNG. The use of the Kalman criterion is therefore added after the recovery of the information by the attacks as a reconstruction of the random number stream.

The purpose of those examples was to show the usefulness of observability for the reconstruction of a random number stream in a RNG attack context. Although very efficient to reconstruct the random number stream, this method still have shortcomings: the dynamic of the CA RNG must be perfectly known; the random number seed (*i.e.* the state of the CA) must not be manually modified by the user during the measurement; the CA RNG must not have non-linearity (which has been recommended in (Meier and Staffelbach, 1989)). By implementing one of the above cases, the application of the Kalman criterion for this CA RNG becomes impossible. The use of a non-linear CA RNG is also possible because the number of cells is large enough to not allow the use of the results of section 3.3.

5.5 Conclusion

In this chapter we were able to apply the observation methods of the thesis to three concrete examples. The first one on a forest fire model, the second on a road traffic model and the third on a random number generator.

In the first example we were able to apply and compare the two observation methods presented in the thesis. The first one using reconstructibility and the second one using synchronisation. The forest fire model chosen was very simple, but the placement of the sensors to ensure reconstructibility and observability was not trivial. It will be necessary to set up methods to place the sensors in order to avoid a trial and error placement which takes a lot of time, especially considering the number of sensors.

The second example was a demonstration of the use of decentralised observability. The example also aimed to illustrate how decentralised observability could be applied to the construction of a modular road traffic network. However, we could not solve this problem for the chosen road traffic model. The verification of decentralised observability or reconstructibility (or more generally of a CA with unknown boundaries) is more time consuming and complex than for other types of boundaries.

The last example served as an illustration for the observability ACA for the observation of a random number generator. In this example, we easily manage to reconstruct the random number sequence from very partial information on the generated numbers. This state estimator can greatly facilitate the exploitation of a security vulnerability. We have presented several suggestions to prevent this kind of attack by making it impossible to use this reconstruction method.

The next and final chapter serves as a conclusion to the thesis. It highlights the main contributions of the thesis and presents some ideas for continuing the work on the observation of CA.

Chapter 6

Conclusions and Perspectives

The observation of natural phenomena is a crucial issue for the understanding of the world around us, especially those with important spatial dynamics. In the introduction, we have seen that cellular automata are good alternatives to PDE to describe spatially distributed systems. This thesis lays the foundations for the observation of CA models, by presenting two profoundly different tools: observability and synchronisation.

Main Contributions

This thesis proposes 4 major contributions: the definition of observability for CA, the generalisation of the Kalman condition for ACA, the adaptation of the observability criterion of BN to CA, and the formalisation of the synchronisation as a state estimator. In the following paragraphs, we will describe each of these contributions in more details.

Firstly, we formalised the notions of observability, reconstructibility and adaptability for cellular automata. Observability and reconstructibility have already been defined for other types of systems (linear time invariant system, boolean network, etc.) in the past. The contribution of this thesis lies in the transposition of these characterisations to cellular automata. We have also described the compared injectivity, a special form of injectivity with respect to another function which is intimately connected to the reconstructibility. Then, we have formalised the notion of adaptability from the work of Laschov *et al.* (2013) to derive a necessary condition for observability that can be checked before the construction of the output sequence. This notion is particularly important as it plays a direct role in the placement of sensors when we wish to ensure observability.

Secondly, we generalised the Kalman rank condition as an observability criterion for additive and affine CA. We proposed similar criteria for adaptability and reconstructibility. These are not a condition on the rank, but respectively on the kernel of the output or observation matrix. The former requires that the kernels be disjoint and the latter requires the inclusion of the kernels. The observability

and reconstructibility criteria are also completed by a corollary that provides the possibility to reconstruct the initial or current state of the system from the output sequence.

Thirdly, we have improved the observability and reconstructibility criteria of Boolean networks by benefiting from the advantages of CA. This allowed us to develop an algorithm that verifies both observability and reconstructibility for each of the initial configurations. Unlike other methods, this algorithm can take into account conditions on the initial configurations. This greatly reduces the algorithmic complexity by providing the ability to evaluate observability or reconstructibility of CA with a large number of cells. In the case where this reduction does not apply or does not reduce the complexity sufficiently, we have implemented a method for decentralising the observability analysis which provides a linearisation of the complexity by dividing the problem into smaller problems. This method works practically only on one-dimensional CA as it requires a large number of sensors for higher dimensions.

Lastly, we have formalised the synchronisation of CA as a state estimator. This approach is radically opposed to the previous one as we do not seek to check beforehand if the observation will work but only to study statistically its performance. We have also developed an optimised method that provides coordination of mobile sensors if the initial error is composed of a single erroneous cell. In the forest fire propagation example, the optimised synchronisation and the reconstructibility approach achieve similar average observation performance.

This thesis lays the foundation for CA observation by providing definitions of observation and two methodologies for state estimation. However, we are still a long way from the deployment of a sensor network using cellular automata for monitoring physical systems. We present hereafter the perspectives to continue this work. We divide them temporally in the order in which we suggest they should be carried out.

Perspectives for Short-term Research

Extension to Continuous State CA

Many CA such as the fire model of Karafyllidis and Thanailakis describe the state of the cells by a continuous value. This type of model has by definition a greater accuracy of modelling than the finite state models. However, they do not fit the definition of El Yacoubi and are therefore not taken into account in this work. The Kalman criterion and the synchronisation still appear to be applicable. The former can be applied as it does not require the finiteness of the state but of the number of variables, *i.e.* that the lattice is finite. The synchronisation can be applied with continuous values but the properties need to be verified.

For the study of non-linear CA, it would be necessary, however, to seek inspiration once again from classical control theory and its study of non-linear systems which has been widely studied for several decades. Another alternative would be to quantify the variable, this would provide, when the state is bounded as in the Karafyllidis and Thanailakis model, a way to use fuzzy logic and thus use the results of this thesis.

Probabilistic Observability and Reconstructibility

At the end of Chapter 3, we introduced probabilistic observability and explained that it might support the use of sub-observers on multidimensional CA. It is necessary to study this since we have simply presented the idea and some definitions of probabilistic observability. Furthermore, establishing criteria for probabilistic observability and reconstructibility would provide a direct way to study probabilistic CA. Although they are less used than deterministic models, they provide a means of modelling features that are impossible for the deterministic one.

Controllability of Cellular Automata

In this thesis we studied the observability and reconstructibility of CA, but not the controllability. Many applications combine the two, hence it is necessary to study this issue. In addition, for linear time invariant systems these two concepts are decoupled, i.e. they can be studied separately. It has already been studied by Dridi *et al.* (2020), but it remains to establish the connection between the two so that a state estimator can be made that is used for system control. The Kalman criterion can be applied in the same way for controllability and it has also been studied for Boolean networks. Bagnoli *et al.* (2012) have also studied synchronisation as a controller.

Perspectives for Intermediate-term Research

Error tolerance

Error tolerance is an essential element of the state estimator. Without it, a measurement error or uncertainty in the model could result in the incorrect reconstruction of the state. In classical control theory, state estimators are constructed to respect error margins that guarantee a certain tolerance to error. In order to apply CA estimation to actual problems, it is necessary to develop these notions of error margins for the estimators presented in this thesis.

Construction of a Robust State Estimator

In this thesis, we have presented the construction of a state estimator. However, the choice in the position of the sensors is not an easy task. It would be necessary to develop methods to automatically place the sensors in such a way as to ensure observability. Adaptability could be used as a condition to do this. The choice of the sensor network and the synthesis of the state estimator must also take into account the error tolerance required for the chosen application.

Perspectives for Long-term Research

Finally, the long-term objective of this research is the application to a real case. This requires all the steps shown in this thesis, from modelling the system by a cellular automaton, to the realisation of a robust state estimator. All this to allow the monitoring of a complex system with a sensor network using a cellular automaton model. In the case of a physical system, this requires additional specific knowledge, for example in engineering for the construction of sensors.

Appendix A

Compared Injectivity

In this appendix we present the notion of compared injectivity, a notion close to classical injectivity. As discussed in section 2.3.3 on page 33, injectivity is to observability what compared injectivity is to reconstructibility. This new notion of injectivity is originally related to the notion of reconstructibility but we have chosen to define it in this appendix to make it independent of the context of the thesis and to allow its use to the widest possible audience.

This appendix is separated in three parts, the first one presents formally the definition of compared injectivity, the second presents different properties, and the last one proposes a method to verify the compared injectivity when considering linear algebra.

In order not to confuse the term injectivity with compared injectivity, we will use the term **usual injectivity** to refer to the classical definition of injectivity. First, we will define some notations that we will use throughout this appendix:

- $f : X \to Y$ the application f from X into Y
- $f(X)$ the codomain of f
- id_X the identity application on X
- Y^X the set of all application $f : X \to Y$

A.1 Definition of Compared Injectivity

While the usual injectivity ensures the uniqueness of the antecedent for the codomain, the **compared injectivity** ensures the uniqueness of another codomain for all elements of the codomain.

Definition A.1 (Compared Injectivity)**.** The application $f : X \to Y$ is **injective with respect** to the application $g : X \to Z$ if and only if :

$$\forall x', x'' \in X, f(x') = f(x'') \implies g(x') = g(x'') \tag{A.1}$$

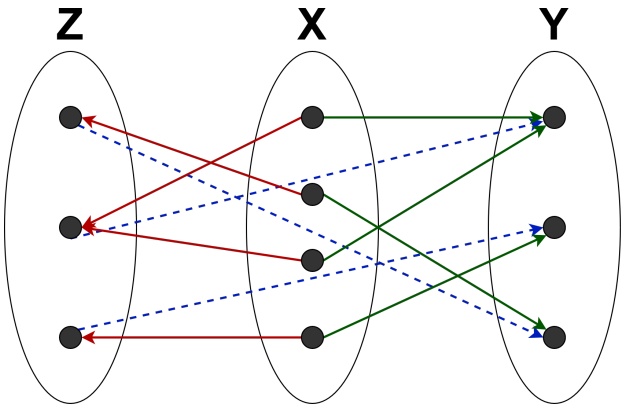


Figure A.1. The sets X, Y and Z are represented by the ellipses with the black dots representing the elements of these sets. The green and red continuous arrows represent the elements associated respectively by the applications f and g. The blue dashed arrows represent the links created by the compared injectivity of f with respect to g.

Directly, we establish that the usual injectivity can be defined as an injectivity with respect to the identity id_X. Moreover, this new formulation allows to formulate an injection between the two codomains, where each element of $f(X)$ is associated with a single element of $g(X)$ as on the Figure A.1. The compared injectivity can thus be expressed as follows:

$$\forall y \in f(X), \exists! z \in Z, \exists x \in X, y = f(x) \text{ and } z = g(x)$$

A.2 Properties of Compared Injectivity

Several properties exist concerning compared injectivity, notably the transitivity property when one of the two functions is usually injective.

Proposition A.1 (Transitivity)**.** *If f is injective with respect to g and g is injective with respect to h, then f is injective with respect to h.*

Proof. Consider the applications $f\colon X \to Y$, $g\colon X \to Z$ and $h\colon X \to W$. Since f is injective with respect to g and g with respect to h then the following two statements are true.

- $\forall x', x'' \in X, f(x') = f(x'') \implies g(x') = g(x'')$
- $\forall x', x'' \in X, g(x') = g(x'') \implies h(x') = h(x'')$

Immediately, it is clear that:

$$\forall x', x'' \in X, f(x') = f(x'') \implies g(x') = g(x'') \implies h(x') = h(x'')$$

We can conclude that f is injective with respect to h. □

Proposition A.2. *If f is **usually injective**, then f is injective with respect to every applications g.*

Proof. Consider the applications $f : X \to Y$ and $g : X \to Z$. If f is usually injective, then f is injective with respect to id_X. Moreover, the following statement is true for all $g \in Y^X$:

$$\forall x', x'' \in X, x' = x'' \implies g(x') = g(x'')$$

Therefore, f is injective with respect to every application $g \in Z^X$. □

Proposition A.3. *If f is injective with respect to g and g is usually injective, then f is usually injective.*

Proof. Consider the applications $f : X \to Y$ and $g : X \to Z$. If g is usually injective, then g is injective with respect to id_X. Using the transitivity property (proposition A.1), if f is injective with respect to g then f is injective with respect to id_X, therefore f is usually injective. □

A.3 Compared Injectivity in Linear Algebra

When we consider that the functions f and g are linear applications, it is possible to check the comparative injectivity more easily.

Proposition A.4. *Let F and G be the matrix of the two linear map f and g. f is injective with respect to g if and only if :*

$$\ker\ F \subset \ker\ G$$

Proof. Consider the matrices F and G of the applications $f : X \to Y$ and $g : X \to Z$. Suppose that f is injective with respect to g, then :

$$\begin{aligned}
&\forall x', x'' \in X, f(x') = f(x'') \implies g(x') = g(x'') \\
\iff &\forall x', x'' \in X, Fx' = Fx'' \implies Gx' = Gx'' \\
\iff &\forall x', x'' \in X, F(x' - x'') = 0 \implies G(x' - x'') = 0 \\
\iff &\forall x', x'' \in X, x' - x'' \in \ker\ F \implies x' - x'' \in \ker\ G \\
\iff &\ker\ F \subseteq \ker\ G
\end{aligned}$$

□

Appendix B

Semi-Tensor Product

In this appendix, we will present in detail the construction of the logic matrix L which describes the dynamics of the Boolean network. We will start by presenting some properties of the Semi-Tensor Product (STP) before applying this method to the BN presented in chapter 3, the example is presented below in Figure B.1.

B.1 Properties of the semi-tensor product

We present here three properties described by Cheng (2005) that will be useful for the construction of the matrix L. The first two will be used as a pseudo-commutativity and the last one will be used to reduce the exponent of a boolean variable.

Remark B.1. It is important to note that the matrices described by Cheng *et al* in their articles are different from ours because of the changes we made as presented as remark 3.2.

The STP has a pseudo-commutativity property when one of the two elements is a vector, this will allow us to isolate the elements of the vector x_t from the elements of the matrix L. It is defined as:

Proposition B.1. *Assume $A \in M_{m\times n}$ is given. Let Z be a column vector of size t. Then*

$$Z \ltimes A = (I_t \otimes A) \ltimes Z \tag{B.1}$$

Next, we define swap matrices, they will be used to have a pseudo commutativity between two vectors. They are defined as:

Definition B.1 (Swap Matrices)**.** The swap matrix $W_{[m,n]}$ is an $mn \times mn$ matrix. Its columns are labelled by $(11, 12, \ldots, 1n, \ldots, m1, m2, \ldots, mn)$ and its rows by $(11, 21, \ldots, m1, \ldots, 1n, 2n, \ldots, mn)$. Then the element in the position $((I, J), (i, j))$ is defined by:

$$w_{(IJ),(ij)} = \begin{cases} 0, & I = i \text{ and } J = j \\ 1, & \text{otherwise} \end{cases}$$

From this matrix, we can define the proposition that allows us to invert two vectors X and Y.

Proposition B.2. *Let X and Y be two column vectors respectively of size m and n.*

$$X \ltimes Y = W_{[n,m]} \ltimes Y \ltimes X \tag{B.2}$$

To calculate L, we will need a reduction matrix M_r which will reduce the exponents of the matrix $M_r A = A^2$. They are defined as :

$$M_r = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{B.3}$$

More generally, Cheng proposes in (Cheng and Qi, 2010, theorem IV.6) to construct the matrix L of the system (B.4) in a systematic way with the equation (B.5).

$$x_{t+1} = M_1 x_t M_2 x_t \dots M_n x_t = L x_t \tag{B.4}$$

$$L = M_1 \prod_{j=2}^{n} \left[(I_{2^n} \otimes M_j) \Phi_n \right] \tag{B.5}$$

where Φ_k is defined by :

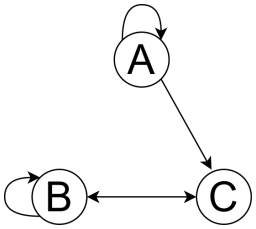
$$\Phi_k = \prod_{i=1}^{k} I_{2^{i-1}} \otimes \left[(I_2 \otimes W_{[2,2^{k-i}]}) M_r \right] \tag{B.6}$$

In the next section, we will calculate the L matrix from the BN shown in Figure B.1.

B.2 Detailed Example on a Simple Boolean Network

In this section, we will give an example of how we can calculate L from the Boolean expressions of the BN. We will use the example presented in chapter 3, it is described in Figure B.1.

First, we will calculate the matrices M_A, M_B and M_C. This is expressed from the logic matrices of the elementary operators which correspond to their truth tables.



(a) Boolean Network

$$\begin{cases} A_{t+1} &= \overline{A_t} = M_A A_t \\ B_{t+1} &= B_t.C_t = M_B B_t C_t \\ C_{t+1} &= \overline{A_t} \oplus B_t = M_C A_t B_t \end{cases}$$

(b) System of equation

FIGURE B.1. Simple example of a three-variable Boolean network described by the graph and the equation system.

$$M_A = M_{\text{not}} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$
$$M_B = M_{\text{and}} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
$$M_C = M_{\text{xor}} M_{\text{not}} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

From the definition of $X_t = A_t B_t C_t$ we can write that :

$$\begin{aligned} X_{t+1} &= A_{t+1} B_{t+1} C_{t+1} \\ &= M_A A_t M_B B_t C_t M_C A_t B_t \\ &= M_A (I_2 \otimes M_B) A_t B_t C_t M_C A_t B_t \\ &= M_A (I_2 \otimes M_B)(I_8 \otimes M_C) A_t B_t C_t A_t B_t \end{aligned}$$

Once the part of the logical matrices is isolated, we can be interested in grouping $A_t B_t C_t A_t B_t$ under the form $A_t B_t C_t$ so as to be able to express X_{t+1} in function of X_t.

$$\begin{aligned} A_t B_t C_t A_t B_t &= A_t B_t W_{[4,2]} A_t B_t C_t \\ &= (I_4 \otimes W_{[4,2]}) A_t B_t A_t B_t C_t \\ &= (I_4 \otimes W_{[4,2]}) A_t W_{[2,2]} A_t B_t B_t C_t \\ &= (I_4 \otimes W_{[4,2]})(I_2 \otimes W_{[2,2]}) A_t A_t B_t B_t C_t \\ &= (I_4 \otimes W_{[4,2]})(I_2 \otimes W_{[2,2]}) M_r A_t M_r B_t C_t \\ &= (I_4 \otimes W_{[4,2]})(I_2 \otimes W_{[2,2]}) M_r (I_2 \otimes M_r) A_t B_t C_t \end{aligned}$$

Finally we have that the matrix L is defined by :

$$L = M_A(I_2 \otimes M_B)(I_8 \otimes M_C)(I_4 \otimes W_{[4,2]})(I_2 \otimes W_{[2,2]})M_r(I_2 \otimes M_r)$$

From definition B.1, we obtain that swap matrices $W_{[2,2]}$ and $W_{[4,2]}$ are :

$$W_{[2,2]} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$W_{[4,2]} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Considering the matrices M_A, M_B and M_C defined above, we obtain that L is:

$$L = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We can check that the matrix L is correct by looking at the adjacency matrix of the graph in Figure Figure 3.3. It is the transpose of the matrix L which confirms that the matrix L is correct.

In this example, we were able to construct the matrix L algebraically. We did the calculations by hand, but using equation (B.5), we can set up an automatic method. However, the number of operations to calculate L is large compared to the number of cells, but these could be optimised with the use of sparse matrices.

Bibliography

Samira El Yacoubi. A mathematical method for control problems on cellular automata models. *International Journal of Systems Science* **39** (5), 529–538 (2008). Cited on page/s vi, vii, xii, 7, 13, 36, 103.

C Chatry, M Le Quentrec, D Laurens, JY Le Gallou, JJ Lafitte, B Creuchet, and J Grelu. Changement climatique et extension des zones sensibles aux feux de forêts. *Rapport de la mission interministérielle CGAAER-CGEDD-IGA, Paris, juillet 2010, 190p* (2010). URL https://www.vie-publique.fr/rapport/31347-changement-climatique-et-extension-des-zones-sensibles-aux-feux-de-foret. Cited on page/s xi, 1.

Rudolf E Kalman. On the general theory of control systems. In *Proceedings First International Conference on Automatic Control, Moscow, USSR* pages 481–492 (1960). Cited on page/s xiii, 7, 86.

Rui-Sheng Wang, Assieh Saadatpour, and Reka Albert. Boolean modeling in systems biology: an overview of methodology and applications. *Physical biology* **9** (5), 055001 (2012). Cited on page/s xiii, 54.

Franco Bagnoli, Samira El Yacoubi, and Raúl Rechtman. Synchronization and control of cellular automata. In *International Conference on Cellular Automata* pages 188–197. Springer (2010). Cited on page/s xiii, 80, 81, 82, 85, 86.

Dmitriy Laschov, Michael Margaliot, and Guy Even. Observability of boolean networks: A graph-theoretic approach. *Automatica* **49** (8), 2351–2362 (2013). Cited on page/s xiv, 8, 27, 34, 36, 54, 59, 60, 61, 62, 137.

Matt J Keeling and Ken TD Eames. Networks and epidemic models. *Journal of the royal society interface* **2** (4), 295–307 (2005). Cited on page/s 1.

Alassane Koné, Allyx Fontaine, and Samira El Yacoubi. Coupling cellular automata with medalus assessment for the desertification issue. In *The International Conference on Emerging Trends in Engineering & Technology (IConETech)* (2020). Cited on page/s 1, 6.

Xinyi Zhou, Atishay Jain, Vir V Phoha, and Reza Zafarani. Fake news early detection: A theory-driven model. *Digital Threats: Research and Practice* **1** (2), 1–25 (2020). Cited on page/s 1.

Richard C Rothermel. A mathematical model for predicting fire spread in wildland fuels volume 115. Intermountain Forest & Range Experiment Station, Forest Service, US … (1972). Cited on page/s 5.

Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. In *World Scientific Reference on Contingent Claims Analysis in Corporate Finance: Volume 1: Foundations of CCA and Equity Valuation* pages 3–21. World Scientific (2019). Cited on page/s 5.

J Hardy, Yves Pomeau, and O De Pazzis. Time evolution of a two-dimensional model system. i. invariant states and time correlation functions. *Journal of Mathematical Physics* **14** (12), 1746–1759 (1973). Cited on page/s 6, 13.

Rafik Ouared and Bastien Chopard. Lattice boltzmann simulations of blood flow: non-newtonian rheology and clotting processes. *Journal of statistical physics* **121** (1), 209–221 (2005). Cited on page/s 6.

B Chopard and M Droz. Cellular automata. *Modelling of Physical* (1998). Cited on page/s 6, 13, 21.

Ioannis Karafyllidis and Adonios Thanailakis. A model for predicting forest fire spreading using cellular automata. *Ecological Modelling* **99** (1), 87–97 (1997). Cited on page/s 6, 14, 30, 102, 103, 104, 105, 107.

Mark A Finney. An overview of flammap fire modeling capabilities. In *In: Andrews, Patricia L.; Butler, Bret W., comps. 2006. Fuels Management-How to Measure Success: Conference Proceedings. 28-30 March 2006; Portland, OR. Proceedings RMRS-P-41. Fort Collins, CO: US Department of Agriculture, Forest Service, Rocky Mountain Research Station. p. 213-220* volume 41 (2006). Cited on page/s 6.

Roland Glowinski and Jacques-Louis Lions. Exact and approximate controllability for distributed parameter systems. *Acta numerica* **4**, 159–328 (1995). Cited on page/s 7.

A El Jai, MC Simon, and E Zerrik. Regional observability and sensor structures. *Sensors and Actuators A: Physical* **39** (2), 95–102 (1993). Cited on page/s 7.

Sara Dridi, Franco Bagnoli, and Samira El Yacoubi. Markov chains approach for regional controllability of deterministic cellular automata, via boundary actions. *Journal of Cellular Automata* **14** (2019). Cited on page/s 8, 74.

Sara Dridi, Samira El Yacoubi, Franco Bagnoli, and Allyx Fontaine. A graph theory approach for regional controllability of boolean cellular automata. *International Journal of Parallel, Emergent and Distributed Systems* **35** (5), 499–513 (2020). Cited on page/s 8, 27, 74, 139.

Jesús Urıas, Gelasio Salazar, and Edgardo Ugalde. Synchronization of cellular automaton pairs. *Chaos: An Interdisciplinary Journal of Nonlinear Science* **8** (4), 814–818 (1998). Cited on page/s 8, 80.

Daizhan Cheng. Semi-tensor product of matrices and its applications to dynamic systems. In *New Directions and Applications in Control Theory* pages 61–79. Springer (2005). Cited on page/s 8, 54, 56, 57, B-1.

John Conway *et al.* The game of life. *Scientific American* **223** (4), 4 (1970). Cited on page/s 12, 16, 27.

Stephen Wolfram. Statistical mechanics of cellular automata. *Reviews of modern physics* **55** (3), 601 (1983). Cited on page/s 13, 18, 19.

Dieter A Wolf-Gladrow. Lattice-gas cellular automata and lattice boltzmann models: an introduction. Springer (2004). Cited on page/s 13, 16.

Parimal Pal Chaudhuri, Dipanwita Roy Chowdhury, Sukumar Nandi, and Santanu Chattopadhyay. Additive cellular automata: theory and applications volume 43. John Wiley & Sons (1997). Cited on page/s 18, 21, 22, 129.

Stephen Wolfram. Universality and complexity in cellular automata. *Physica D: Nonlinear Phenomena* **10** (1-2), 1–35 (1984). Cited on page/s 18, 27.

Norman H Packard and Stephen Wolfram. Two-dimensional cellular automata. *Journal of Statistical physics* **38** (5), 901–946 (1985). Cited on page/s 18.

Matthew Cook *et al.* Universality in elementary cellular automata. *Complex systems* **15** (1), 1–40 (2004). Cited on page/s 20, 27.

Olivier Martin, Andrew M Odlyzko, and Stephen Wolfram. Algebraic properties of cellular automata. *Communications in mathematical physics* **93** (2), 219–258 (1984). Cited on page/s 21, 26.

Marco Tomassini, Moshe Sipper, and Mathieu Perrenoud. On the generation of high-quality random numbers by two-dimensional cellular automata. *IEEE Transactions on computers* **49** (10), 1146–1151 (2000). Cited on page/s 21, 24, 129, 130, 131.

Pabitra Pal Choudhury, Sudhakar Sahoo, Mithun Chakraborty, Subir Kumar Bhandari, and Amita Pal. Investigation of the global dynamics of cellular automata using boolean derivatives. *Computers & Mathematics with Applications* **57** (8), 1337–1351 (2009). Cited on page/s 24.

Jean Mairesse and Irene Marcovici. Around probabilistic cellular automata. *Theoretical Computer Science* **559**, 42–72 (2014). Cited on page/s 25.

Edward F Moore. Machine models of self-reproduction. In *Proceedings of symposia in applied mathematics* volume 14 pages 17–33. American Mathematical Society New York (1962). Cited on page/s 28.

John Myhill. The converse of moore's garden-of-eden theorem. *Proceedings of the american mathematical society* **14** (4), 685–686 (1963). Cited on page/s 28.

Rudolf Emil Kalman. Mathematical description of linear dynamical systems. *Journal of the Society for Industrial and Applied Mathematics, Series A: Control* **1** (2), 152–192 (1963). Cited on page/s 33, 40.

Théo Plénet, Samira El Yacoubi, Clément Raïevsky, and Laurent Lefèvre. Observability of affine cellular automaton through mobile sensors. In *International Conference on Cellular Automata for Research and Industry* pages 36–45. Springer (2020a). Cited on page/s 37.

Théo Plénet, Samira El Yacoubi, Clément Raïevsky, and Laurent Lefèvre. Observability and reconstructibility of bounded cellular automata. *International Journal of Systems Science* pages 1–17 (2022a). Cited on page/s 37, 78.

PE Sarachik and E Kreindler. Controllability and observability of linear discrete-time systems. *International Journal of Control* **1** (5), 419–432 (1965). Cited on page/s 40.

Adi Ben-Israel and Thomas NE Greville. Generalized inverses: theory and applications volume 15. Springer Science & Business Media (2003). Cited on page/s 45.

Athanasios C Antoulas. Approximation of large-scale dynamical systems. SIAM (2005). Cited on page/s 49.

Roger Penrose. On best approximate solutions of linear matrix equations. In *Mathematical Proceedings of the Cambridge Philosophical Society* volume 52 pages 17–19. Cambridge University Press (1956). Cited on page/s 50.

Robert M Gray. Toeplitz and circulant matrices: A review. now publishers inc (2006). Cited on page/s 52.

Ho Yee Cheung, Tsz Chiu Kwok, and Lap Chi Lau. Fast matrix rank algorithms and applications. *Journal of the ACM (JACM)* **60** (5), 1–25 (2013). Cited on page/s 53.

Daizhan Cheng, Hongsheng Qi, and Zhiqiang Li. Analysis and control of boolean networks: a semi-tensor product approach. Springer Science & Business Media (2010). Cited on page/s 54.

Stuart A Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology* **22** (3), 437–467 (1969). Cited on page/s 54.

Réka Albert and Albert-László Barabási. Dynamics of complex systems: Scaling laws for the period of boolean networks. *Physical Review Letters* **84** (24), 5660 (2000). Cited on page/s 54.

Ilya Shmulevich, Edward R Dougherty, and Wei Zhang. From boolean to probabilistic boolean networks as models of genetic regulatory networks. *Proceedings of the IEEE* **90** (11), 1778–1792 (2002). Cited on page/s 54.

Daizhan Cheng and Hongsheng Qi. Controllability and observability of boolean control networks. *Automatica* **45** (7), 1659–1667 (2009). Cited on page/s 54, 56, 57, 58, 59, 60, 62.

Carlos Gershenson. Classification of random boolean networks. *arXiv preprint cs/0208001* (2002). Cited on page/s 55.

Daizhan Cheng and Hongsheng Qi. Matrix expression of logic and fuzzy control. In *Proceedings of the 44th IEEE Conference on Decision and Control* pages 3273–3278. IEEE (2005). Cited on page/s 56, 57.

Daizhan Cheng and Hongsheng Qi. A linear representation of dynamics of boolean networks. *IEEE Transactions on Automatic Control* **55** (10), 2251–2258 (2010). Cited on page/s 57, B-2.

Raphaël M Jungers and Vincent D Blondel. Observable graphs. *Discrete Applied Mathematics* **159** (10), 981–989 (2011). Cited on page/s 61.

Aydin Buluç, Jeremy T Fineman, Matteo Frigo, John R Gilbert, and Charles E Leiserson. Parallel sparse matrix-vector and matrix-transpose-vector multiplication using compressed sparse blocks. In *Proceedings of the twenty-first annual symposium on Parallelism in algorithms and architectures* pages 233–244 (2009). Cited on page/s 63.

Emmanuel N Millán, Nicolás Wolovick, Maria Fabiana Piccoli, Carlos Garcia Garino, and Eduardo M Bringa. Performance analysis and comparison of cellular automata gpu implementations. *Cluster Computing* **20** (3), 2763–2777 (2017). Cited on page/s 63.

Junqi Yang, Wei Qian, and Zhiqiang Li. Redefined reconstructibility and state estimation for boolean networks. *IEEE Transactions on Control of Network Systems* 7 (4), 1882–1890 (2020). Cited on page/s 67.

Ulisses Braga-Neto. Optimal state estimation for boolean dynamical systems. In *2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)* pages 1050–1054. IEEE (2011). Cited on page/s 67.

Yuqian Guo, Qunming Li, and Weihua Gui. Optimal state estimation of boolean control networks with stochastic disturbances. *IEEE Transactions on Cybernetics* **50** (3), 1355–1359 (2018). Cited on page/s 71.

Kuize Zhang and Karl Henrik Johansson. Efficient verification of observability and reconstructibility for large boolean control networks with special structures. *IEEE Transactions on Automatic Control* **65** (12), 5144–5158 (2020). Cited on page/s 72.

Giuseppe Basile and Giovanni Marro. On the observability of linear, time-invariant systems with unknown inputs. *Journal of Optimization theory and applications* **3** (6), 410–415 (1969). Cited on page/s 73.

A Bel Fekih and A El Jai. Regional analysis of a class of cellular automata models. In *International conference on cellular automata* pages 48–57. Springer (2006). Cited on page/s 73.

Ettore Fornasini and Maria Elena Valcher. Observability and reconstructibility of probabilistic boolean networks. *IEEE Control Systems Letters* **4** (2), 319–324 (2019). Cited on page/s 76.

Théo Plénet, Samira El Yacoubi, Clément Raïevsky, and Laurent Lefèvre. Observability and reconstructibility of affine cellular automata: Example on random number reconstruction. *Journal of Cellular Automata* (2022b). in press. Cited on page/s 78.

Alexander L Fradkov and Boris Andrievsky. Synchronization and phase relations in the motion of two-pendulum system. *International Journal of Non-Linear Mechanics* **42** (6), 895–901 (2007). Cited on page/s 80.

Radu Dogaru, Ioana Dogaru, and Hyongsuk Kim. Binary chaos synchronization in elementary cellular automata. *International Journal of Bifurcation and Chaos* **19** (09), 2871–2884 (2009). Cited on page/s 80.

Franco Bagnoli and Raúl Rechtman. Synchronization and maximum lyapunov exponents of cellular automata. *Physical Review E* **59** (2), R1307 (1999). Cited on page/s 80.

Jürgen Ackermann. Der entwurf linearer regelungssysteme im zustandsraum. *at-Automatisierungstechnik* **20** (1-12), 297–300 (1972). Cited on page/s 86.

Rudolf Emil Kalman *et al.* Contributions to the theory of optimal control. *Bol. soc. mat. mexicana* **5** (2), 102–119 (1960). Cited on page/s 86.

Théo Plénet, Clément Raïevsky, Laurent Lefèvre, and Samira El Yacoubi. Social organisation of mobile sensors for wildfire spread estimation. *IFAC-PapersOnLine* **53** (2), 3596–3601 (2020b). Cited on page/s 99, 120.

A Hernández Encinas, L Hernández Encinas, S Hoya White, A Martín del Rey, and G Rodríguez Sánchez. Simulation of forest fire fronts using cellular automata. *Advances in Engineering Software* **38** (6), 372–378 (2007a). Cited on page/s 105.

L Hernández Encinas, S Hoya White, A Martín Del Rey, and G Rodríguez Sánchez. Modelling forest fire spread using hexagonal cellular automata. *Applied mathematical modelling* **31** (6), 1213–1227 (2007b). Cited on page/s 105.

Stephen G Berjak and John W Hearne. An improved cellular automaton model for simulating fire in a spatially heterogeneous savanna system. *Ecological modelling* **148** (2), 133–151 (2002). Cited on page/s 105.

Holly A Perryman, Christopher J Dugaw, J Morgan Varner, and Diane L Johnson. A cellular automata model to link surface fires to firebrand lift-off and dispersal. *International journal of wildland fire* **22** (4), 428–439 (2012). Cited on page/s 105.

A Alexandridis, L Russo, D Vakalis, GV Bafas, and CI Siettos. Wildland fire spread modelling using cellular automata: evolution in large-scale spatially heterogeneous environments under fire suppression tactics. *International Journal of Wildland Fire* **20** (5), 633–647 (2011). Cited on page/s 105.

Rodolfo Maduro Almeida and Elbert EN Macau. Stochastic cellular automata model for wildland fire spread dynamics. In *Journal of Physics: Conference Series* volume 285 page 012038. IOP Publishing (2011). Cited on page/s 105.

Giuseppe A Trunfio, Donato D'Ambrosio, Rocco Rongo, William Spataro, and Salvatore Di Gregorio. A new algorithm for simulating wildfire spread through cellular automata. *ACM Transactions on Modeling and Computer Simulation (TOMACS)* **22** (1), 1–26 (2011). Cited on page/s 107.

Guillaume Schlotterbeck, Clement Raïevsky, and Laurent Lefèvre. Decentralized estimation of forest fire spread using mobile sensors. In *International Conference on Cellular Automata* pages 334–343. Springer (2016). Cited on page/s 107, 114, 120.

Suparna Chakraborty, Neeraj Kumar Goyal, Sudipta Mahapatra, and Sieteng Soh. A monte-carlo markov chain approach for coverage-area reliability of mobile wireless sensor networks with multistate nodes. *Reliability Engineering & System Safety* **193**, 106662 (2020). Cited on page/s 114.

Charles C Castello, Jeffrey Fan, Asad Davari, and Ruei-Xi Chen. Optimal sensor placement strategy for environmental monitoring using wireless sensor networks. In *2010 42nd Southeastern Symposium on System Theory (SSST)* pages 275–279. IEEE (2010). Cited on page/s 114.

Wei Liu, Wei-cheng Gao, Yi Sun, and Min-jian Xu. Optimal sensor placement for spatial lattice structure based on genetic algorithms. *Journal of Sound and Vibration* **317** (1-2), 175–189 (2008). Cited on page/s 114.

Michael A Demetriou and Islam I Hussein. Estimation of spatially distributed processes using mobile spatially distributed sensor network. *SIAM Journal on Control and Optimization* **48** (1), 266–291 (2009). Cited on page/s 114.

Michael A Demetriou. Guidance of mobile actuator-plus-sensor networks for improved control and estimation of distributed parameter systems. *IEEE Transactions on Automatic Control* **55** (7), 1570–1584 (2010). Cited on page/s 114.

Kai Nagel and Michael Schreckenberg. A cellular automaton model for freeway traffic. *Journal de physique I* **2** (12), 2221–2229 (1992). Cited on page/s 122.

Wolfgang Knospe, Ludger Santen, Andreas Schadschneider, and Michael Schreckenberg. A realistic two-lane traffic model for highway traffic. *Journal of Physics A: Mathematical and General* **35** (15), 3369 (2002). Cited on page/s 122.

Elmar Brockfeld, Robert Barlovic, Andreas Schadschneider, and Michael Schreckenberg. Optimizing traffic lights in a cellular automaton model for city traffic. *Physical review E* **64** (5), 056132 (2001). Cited on page/s 122.

Sven Maerivoet and Bart De Moor. Cellular automata models of road traffic. *Physics reports* **419** (1), 1–64 (2005). Cited on page/s 122.

Joachim Krug and Herbert Spohn. Universality classes for deterministic surface growth. *Physical Review A* **38** (8), 4271 (1988). Cited on page/s 122.

David A Rosenblueth and Carlos Gershenson. A model of city traffic based on elementary cellular automata. *Complex Systems* **19** (4), 305 (2011). Cited on page/s 122, 127, 129.

Kohei Higashi, Junkichi Satsuma, and Tetsuji Tokihiro. Rule 184 fuzzy cellular automaton as a mathematical model for traffic flow. *Japan Journal of Industrial and Applied Mathematics* **38** (2), 579–609 (2021). Cited on page/s 122.

Katsuhiro Nishinari and Daisuke Takahashi. Analytical properties of ultradiscrete burgers equation and rule-184 cellular automaton. *Journal of Physics A: Mathematical and General* **31** (24), 5439 (1998). Cited on page/s 122.

Kinga Marton, Alin Suciu, and Iosif Ignat. Randomness in digital cryptography: A survey. *Romanian journal of information science and technology* **13** (3), 219–240 (2010). Cited on page/s 129.

George Marsaglia. Diehard: a battery of tests of randomness. *http://stat. fsu. edu/geo* (1996). Cited on page/s 129, 130.

Pierre L'ecuyer and Richard Simard. Testu01: Ac library for empirical testing of random number generators. *ACM Transactions on Mathematical Software (TOMS)* **33** (4), 1–40 (2007). Cited on page/s 129.

John Kelsey, Bruce Schneier, David Wagner, and Chris Hall. Cryptanalytic attacks on pseudorandom number generators. In *International workshop on fast software encryption* pages 168–188. Springer (1998). Cited on page/s 129.

Peter D Hortensius, Robert D Mcleod, Werner Pries, D Michael Miller, and Howard C Card. Cellular automata-based pseudorandom number generators for built-in self-test. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **8** (8), 842–859 (1989). Cited on page/s 129.

Ph Tsalides, TA York, and A Thanailakis. Pseudorandom number generators for vlsi systems based on linear cellular automata. *IEE Proceedings E (Computers and Digital Techniques)* **138** (4), 241–249 (1991). Cited on page/s 129.

David Samyde, Sergei Skorobogatov, Ross Anderson, and J-J Quisquater. On a new way to read data from memory. In *First International IEEE Security in Storage Workshop, 2002. Proceedings.* pages 65–69. IEEE (2002). Cited on page/s 132.

François-Xavier Standaert. Introduction to side-channel attacks. In *Secure integrated circuits and systems* pages 27–42. Springer (2010). Cited on page/s 132, 133.

Willi Meier and Othmar Staffelbach. Nonlinearity criteria for cryptographic functions. In *Workshop on the Theory and Application of of Cryptographic Techniques* pages 549–562. Springer (1989). Cited on page/s 134.

Franco Bagnoli, Raúl Rechtman, and Samira El Yacoubi. Control of cellular automata. *Physical Review E* **86** (6), 066201 (2012). Cited on page/s 139.