Introduction aux automates cellulaires, aspects calculatoires et universalité

Xavier Provençal

22 octobre 2024

Table des matières

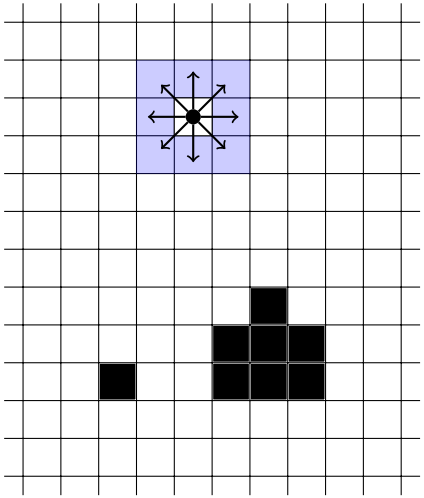
Table des matières

Table des matières

Le jeu de la vie

Le *jeu de la vie* est un *automate cellulaire* inventé par John Conway en 1970.

- On considère un grille infinie dont chaque case est appelée une **cellule**.
- Chaque cellule est soit [Vivante ■ / Morte □]
- Chaque cellule possède 8 **voisines**.
- Règles :
  - Une cellule vivante reste vivante si elle a 2 ou 3 voisines vivantes, sinon elle meurt.
  - Une cellule morte devient vivante si elle a exactement 3 voisines vivantes.
- Le temps est discrétisé en **générations**.



Génération : 01234

Table des matières

Automate cellulaire

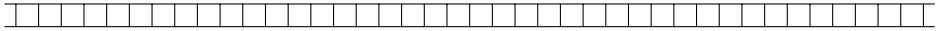
Pour construire un *automate cellulaire*, on doit spécifier quatre chose :

1. un *réseau* de cellules,
2. un ensemble fini d'*états* possibles pour les cellules,
3. une notion de *voisinage*,
4. une *fonction de transition*.

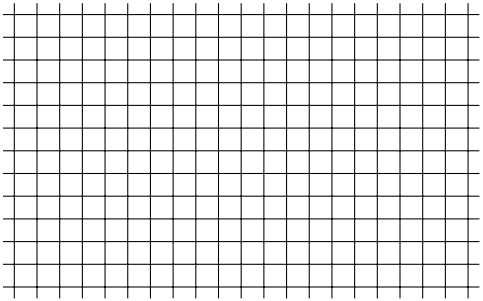
1. Le réseau

Le **réseau** d'un automate cellulaire est une grille régulière de dimension d dont chaque case est appelée une **cellule**.

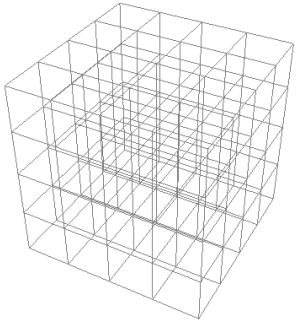
$d = 1$



$d = 2$:



$d = 3$:

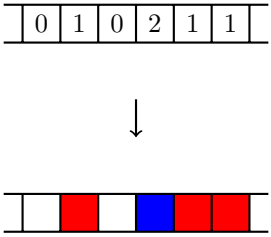


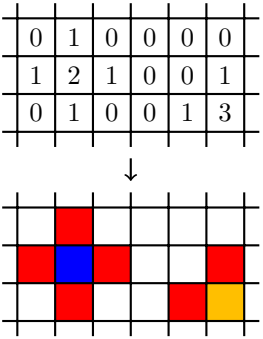
2. Les états

À tout moment, chaque cellule est dans un et un seul **état** choisi parmi un **ensemble d'états** fini.

Habituellement,

- il y a n états et l'ensemble des états est $\{0, 1, \ldots, n-1\}$;
- habituellement, chaque état est représenté par une couleur.

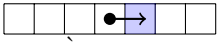




3. Voisinage

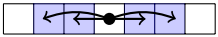
Chaque cellule possède un **voisinage**, c'est-à-dire un ensemble de cellules qui sont considérées comme ses voisines.

- Exemples en dimension 1,



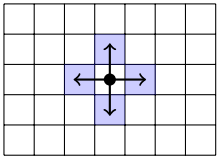
À droite

Rayon 1

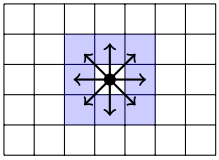


Rayon 2

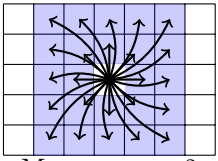
- Exemples en dimension 2,



Von Neumann



Moore



Moore, rayon 2

4. Fonction de transition

La **fonction de transition** est la règle qui donne le nouvel état d'une cellule en fonction de son état actuel et des états de ses voisines.

Exemple :

- Dimension 1, états : $\{\underset{0}{\square}, \underset{1}{\blacksquare}\}$, voisinage : rayon 1, fonction de transition :

- Génération 0 :
- Génération 1 :
- Génération 2 :
- Génération 3 :
- Génération 4 :

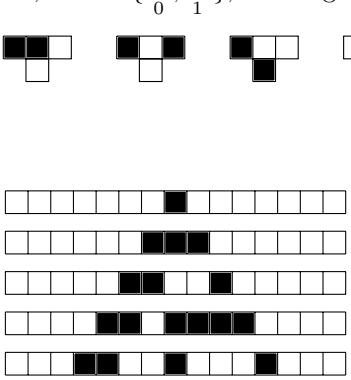


Diagramme espace-temps :

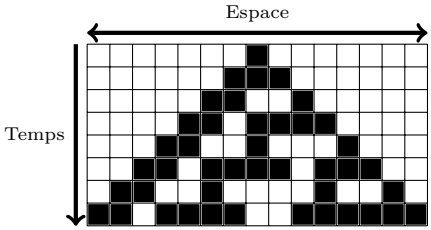
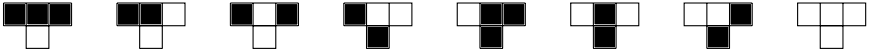


Diagramme espace-temps

Exemple :

- Dimension 1, états : { □, ■ }, voisinage : rayon 1, fonction de transition :



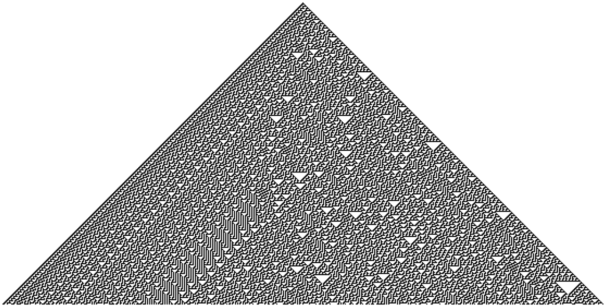


Diagramme espace-temps.

Conus textile (mollusque).

Table des matières

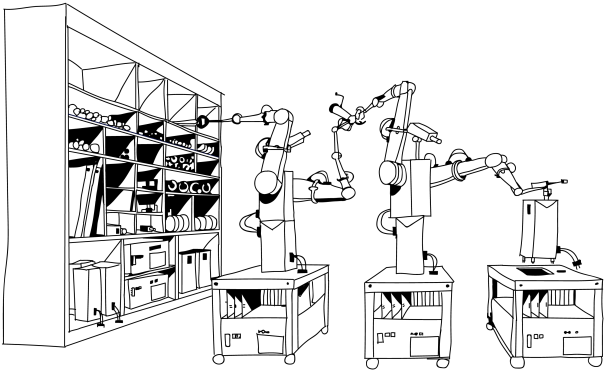
Von Neumann et les machines auto-réplicante

Dans les années 1940, John von Neumann s'intéresse à la colonisation de Mars.

$$Fe_2O_3 \rightarrow Fe + O$$

Mars

John von Neumann



Solution : y envoyer une petite machine auto-réplicante.

Sous les conseils de Stanisław Ulam, il *invent* le concept d'automate cellulaire et en conçoit un exemple 2D, à 29 états dans lequel un motif précis est capable de se *répliquer*.

Plusieurs points de vues sur les automates cellulaires

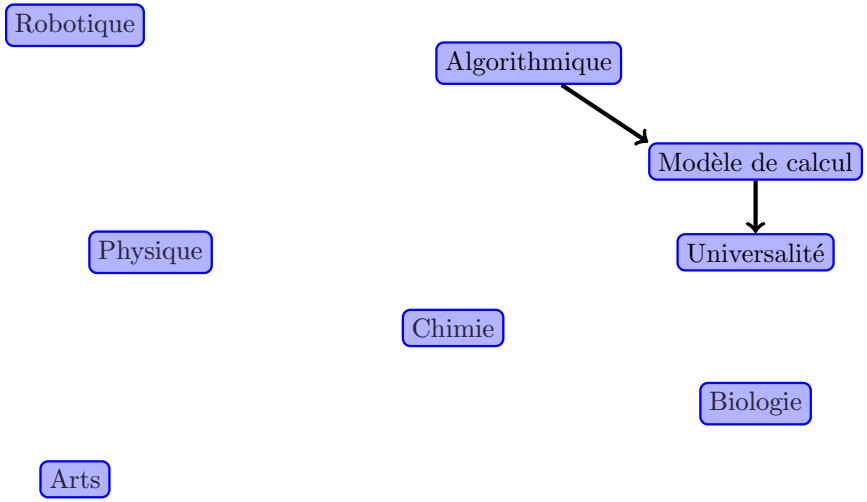
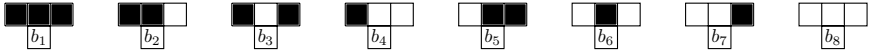


Table des matières

Le cas le plus simple...

Automates cellulaires élémentaires : dimension 1, états : $\{\underset{0}{\square}, \underset{1}{\blacksquare}\}$, voisinage : rayon 1.

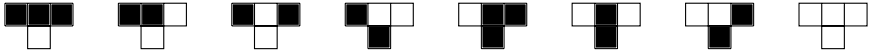
Un tel AC est entièrement déterminé par sa fonction de transition



Un AC est donc déterminé par 8 bits d'information : $b_1, b_2, \cdots, b_8$.

Il y a donc $2^8 = 256$ règles possibles, identifiées par les nombres de 0 à 255.

L'exemple de tout à l'heure est la **règle 30**.



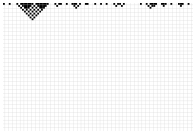
$$(00011110)_2 = 30.$$

Outil de visualisation : ici.

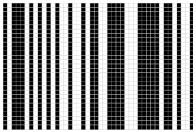
Classification de Wolfram

À partir d'expérimentations, Wolfram sépare les AC en quatre classes :

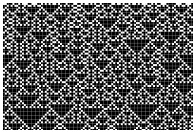
- Classe 1 : tous les états initiaux évoluent vers le même point fixe.



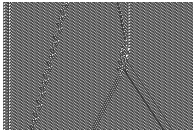
- Classe 2 : plusieurs états finaux possibles, mais tous sont périodiques.



- Classe 3 : apparence de chaos, mais des structures locales apparaissent.



- Classe 4 : mélange d'ordre et de chaos où des structures simples se déplacent en interagissent entre elles.



Prédire le comportement d'un automate cellulaire

Soit $\mathcal{A}$ un AC doté d'un état quiescent quel est le *comportenement* de $\mathcal{A}$ sur les motifs finis ?

- Étant donné M et k, peut-on *calculer* M' sans simuler les k générations ?
- Étant donné M, existe-t-il k tel que M' est stable ?
- Existe-t-il des motifs M tel que pour tout k, M' est non nul ?

Il **ne peut pas exister** de programme informatique qui répond correctement à une de ces questions pour tout AC.

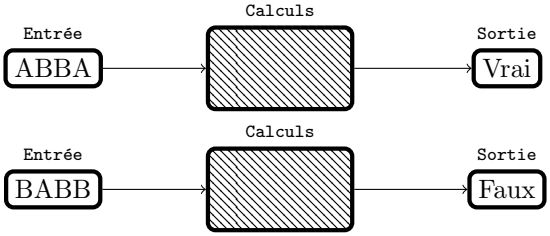
Table des matières

Modèle de calcul

En informatique théorique, un **modèle de calcul** est une construction mathématique qui :

- prend en entrée une suite de symboles appelée les **données** du problème ;
- effectue un *calcul* sur ces données ;
- si le calcul termine, produit en sortie une **réponse**.

Exemple : le problème de reconnaître les palindromes.



Exemples de calculs par AC

Il existe des AC capable de :

- Reconnaître les palindromes.
- Énumérer les nombres premiers (Fischer, 1965).
- Simuler un ordinateur...

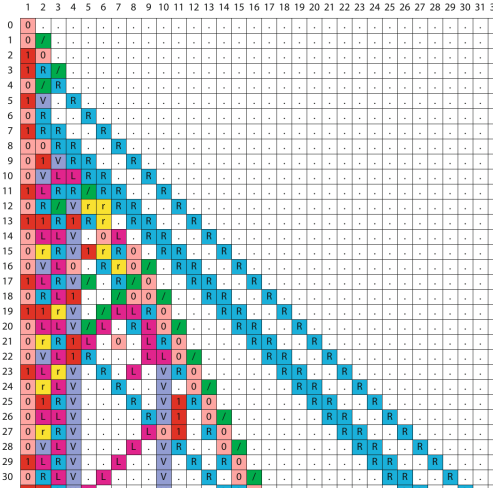
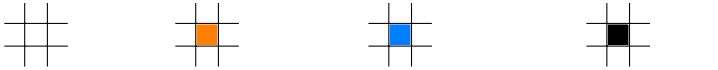


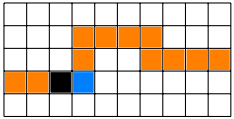
Diagramme espace-temps d'un automate cellulaire à 8 états dont la cellule 0 faut 1 à la génération k ssi k est premier.
(Source : Umeo, Moyamoto et Abe, *Real-Time Prime Generators Implemented on Small-State Cellular Automata*, Chap. 15 du livre *Automata, Universality, Computation*, 2015.)

Wireworld est un AC 2D créé par Brian Silverman en 1987.

- Quatre états :



- La fonction de transition simule le *déplacement* d'*électrons* dans les fils de cuivre.



Génération 3

- Ces règles permettent de construire des portes logiques, des diodes, des flip-flops, etc.

Universalité des AC

L'AC Wireworld est capable de reproduire l'exécution d'un ordinateur qui exécute un programme quelconque.

Théorème (Turing, 1936)

Le *problème de l'arrêt* est *indécidable*.

- **Problème de l'arrêt** : étant donné le code d'un programme informatique, est-ce que son exécution termine ?
- **Indécidable** : il ne peut pas existe de programme informatique qui produit une réponse correcte pour tous les cas.

Ainsi, il ne peut pas exister de programme informatique qui détermine si un automate cellulaire quelconque atteindra un état stable ou non.

Universalité de Turing

Un AC capable de simuler l'exécution d'un ordinateur est dit **Turing-universel**.

Est-ce que l'universalité de Wireworld est une exception ?

- Jeu de la vie
  - En 1982, Berlekamp, Conway et Guy montrent que le jeu de la vie est Turing-universel.
  - En 2012, Nicholas Carlini construit émulateur du Intel 4004 dans le jeu de la vie (lien).
- En 2004, Matthew Cook montre l'AC élémentaire, **règle 110** est Turing-universel.

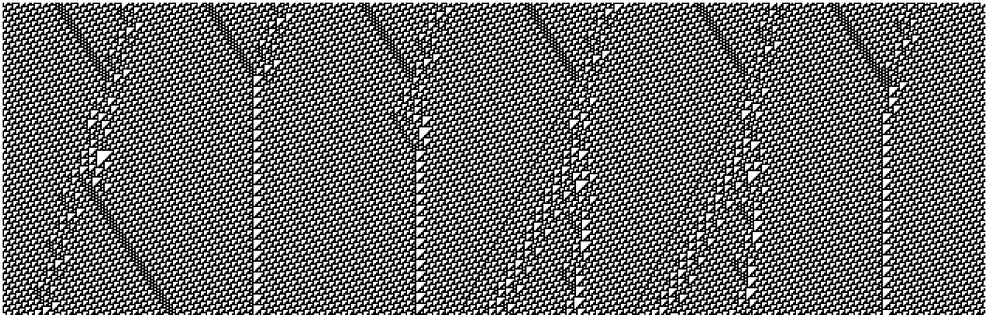
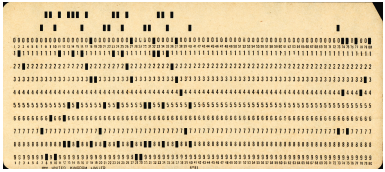


Table des matières

Table des matières

L'informatique avant les ordinateurs

Des machines à traîter l'information existaient bien avant les ordinateurs.



Carte perforée

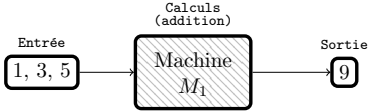
Machine à traiter l'information utilisée dans le cadre du recensement de 1890 aux États-Unis.

(Source : Google Arts & Culture, *Punched Card Machines*)

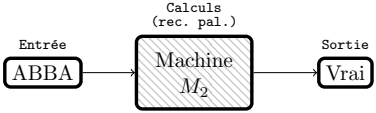
Machines de Turing

En 1936, Alan Turing propose un modèle de machine théorique à la fois simple et puissant. Selon la configuration choisi, la machine peut :

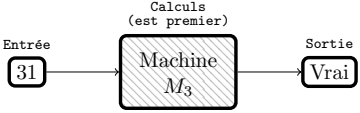
- Calculer des fonctions mathématiques.



- Reconnaître les palindrômes.

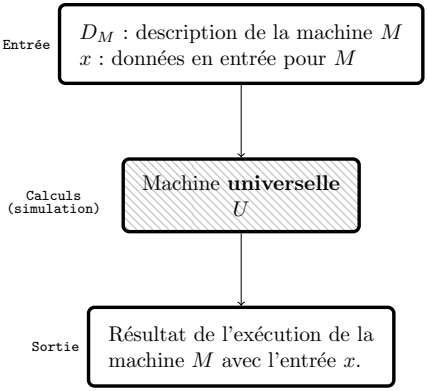


- Reconnaître les nombres premiers.



Machine de Turing universelle

Turing montre qu'il existe une machine capable de simuler n'importe quelle autre machine de Turing.



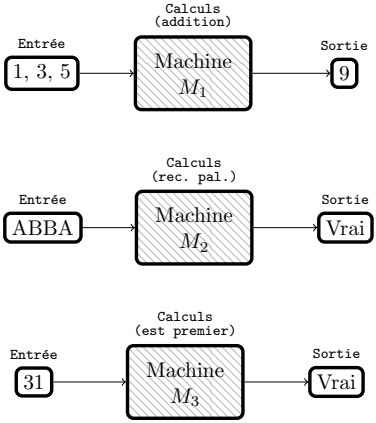


Table des matières

Universalité intrinsèque et extrinsèque

- **Universalité intrinsèque**
  - Capacité à simuler toutes les instances de son propre modèle de calcul.
  - Exemple : la machine de Turing U simule toutes les machines de Turing.

- **Universalité extrinsèque**
  - Capacité à simuler toutes les instances d'un autre modèle de calcul.
  - Exemple : l'AC *jeu de la vie* peut simuler toutes les machine de Turing (tous les programmes informatiques).

Table des matières

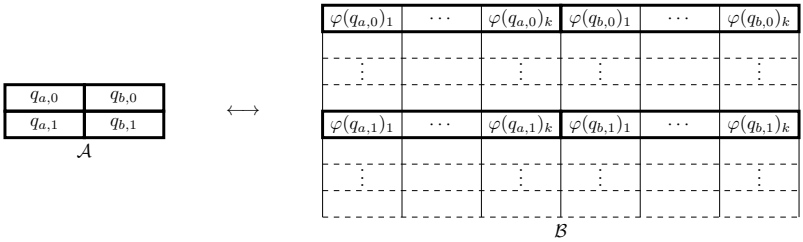
Universalité des AC

- Dans son livre de 1966, von Neumann définit un AC Turing-Universel. (Universalité extrinsèque)
- Il n'y a pas de consensus autour d'une définition formelle d'**universalité intrinsèque** des AC. Voir Ollinger (2008).
- Dans son livre de 1984, Stephen Wolfram donne un exemple d'automate cellulaire 1D, à 19 états et un voisinage de rayon 2 qu'il qualifie d'*universel*.

Une manière de définir le fait que $\mathcal{A}$ **simule** $\mathcal{B}$ est,

$$\begin{array}{ccccc} & & C & \xrightarrow{\overline{\varphi}} & \overline{\varphi}(C) \\ \mathcal{A} \leq \mathcal{B} & \iff & f_{\mathcal{A}} \downarrow & & \downarrow f^*_{\mathcal{B}} \\ & & f_{\mathcal{A}}(C) & \xrightarrow{\overline{\varphi}} & \overline{\varphi}(f^*_{\mathcal{A}}(C)) \end{array}$$

où $f_{\mathcal{A}}, f_{\mathcal{B}}$ sont les fonctions de transition et φ est une fonction injective des états de $\mathcal{A}$ vers des blocs d'états de $\mathcal{B}$.



Fréquence de l'universalité intrinsèque

- Un automate $\mathcal{B}$ est dit **intrinsèquement universel** si $\forall \mathcal{A}, \quad \mathcal{A} \leq \mathcal{B}$.

- Boyer et Theyssier (2009) fournissent des résultats *probabilistes* sur l'universalité intrinsèque.

5. Universality Everywhere

Gathering the density results of section 3 and the existential results for universality in section 4, we obtain an asymptotic density 1 for universality in the following classes.

Family $\mathcal{F}$	**Condition on the path ρ**	**Comments**
Captive CA	$\exists k_0$ s.t. $\rho(x) = (x, k_0)$	Already in [9]
Multiset CA	$\exists n_0$ s.t. $\rho(x) = (n_0, x)$	
k'-outer-multiset	$\exists n_0$ s.t. $\rho(x) = (n_0, x)$	$k' = o(\log(\log k))$
Totalistic CA	$\exists n_0$ s.t. $\rho(x) = (n_0, x)$	
k'-outer-totalistic	$\exists n_0$ s.t. $\rho(x) = (n_0, x)$	$k' = o(\log(\log k))$
Set captive CA	$\underline{\lim}_{x\to\infty} \pi_1(\rho(x)) = +\infty$	
Multiset captive CA	$\underline{\lim}_{x\to\infty} \pi_1(\rho(x)) = +\infty$	

Table des matières

Références (1/2)

- Logiciel de visualisation des automates cellulaires :
  - Golly : http://golly.sourceforge.net/
- Sites web :
  - Golly, version web : https://golly.sourceforge.io/webapp/golly.html
  - Automates cellulaires binaires 1D : https://devinacker.github.io/celldemo/
  - Émulation d'un ordinateur dans WireWorld : https://www.quinapalus.com/wi-index.html
  - Émulation d'un ordinateur dans le jeu de la vie : https://devinacker.github.io/celldemo/
  - Google Arts & Culture, *Punched Card Machines* : https://artsandculture.google.com/story/punched-card-machines-the-national-museum-of-computing/bwWBrooyeGKPiA
- Exposés :
  - Ollinger, *Tutorial on cellular automata*, Lecture 1, Lecture 2, Research School in Discrete Mathematics and Computer Science, CIRM, 2024.

Références (2/2)

- Bibliographie
  - Portrait global (livres)

    [1] Adamatzky (ed.), *Automata, Universality, Computation – Tribute to Maurice Margenstern*, ECC, Springer, 2015.
    [2] Wolfram, *A new kind of science*, Wolfram media, 2002.

  - Identification des nombres premiers

    [3] Fischer, *Generation of Primes by a One-Dimensional Real-Time Cellular Automaton*, Journal of ACM, 1965.
    [4] Korec, *Real-Time Generation of Primes by a One-Dimensional Cellular Automaton with 11 States*, Math. Found. of Computer Science, 1997.

  - Auto-réplication

    [5] Langton, *Self-reproduction in cellular automata*, Physica D : Nonlinear Phenomena, 1984.
    [6] Sayama et Nehaniv *Self-Reproduction and Evolution in Cellular Automata : 25 Years after Evoloops*, CoRR, 2024.
    [7] Von Neumann, *Theory of Self-Reproducting Automata*, University of Illinois Press, 1966.

  - Les différentes universalités des AC

    [8] Ollinger, *Universalities in cellular automata ; a (short) survey*, JAC, 2008.

  - Universalité au sens de Turing des AC

    [9] Berlekamp, Conway et Guy, *What is life*, chapitre 25 de *Winning Ways for your Mathematical Plays*, Academic Press, 1982.
    [10] Cook, *Universality in Elementary Cellular Automata*, Complex Systems, 2004.
    [11] Rendell, *Turing Machine Universality of the Game of Life*, ECC, 2016.

  - Universalité intrinsecte

    [12] Boyer, *Comportement typiques dans les automates cellulaires*, Thèse de doctorat, 2010.
    [13] Boyer et Theyssier, *On local symmetries and Universality in Cellular Automata*, Symposium on Theoretical Aspects of Computer Science, 2009.
    [14] Delorme, Mazoyer, Ollinger et Theyssier, *Bulking I : An abstract theory of bulking*, Theoretical Computer Science, 2011.
    [15] Delorme, Mazoyer, Ollinger et Theyssier, *Bulking II : Classification of cellular automata*, Theoretical Computer Science, 2011.
    [16] Mazoyer and Rapaport, *Inducing an Order on Cellular Automata by a Grouping Operation*. STACS, 1998.
    [17] Ollinger, *Automates cellulaires : structures*, Thèse de doctorat, 2002.
    [18] Theyssier, *Automates cellulaires : un modèle de complexité*, Thèse de doctorat, 2005.
