



Conjunctive grammars, cellular automata and logic

Théo Grente, Etienne Grandjean

► **To cite this version:**

Théo Grente, Etienne Grandjean. Conjunctive grammars, cellular automata and logic. 2021. hal-03167529

HAL Id: hal-03167529

<https://hal.archives-ouvertes.fr/hal-03167529>

Preprint submitted on 12 Mar 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Conjunctive grammars, cellular automata and logic

Théo Grente ✉

GREYC, Université de Caen Normandie, France

Étienne Grandjean ✉

GREYC, Université de Caen Normandie, France

Abstract: The expressive power of the class Conj of *conjunctive languages*, i.e. languages generated by the *conjunctive grammars* of Okhotin, is largely unknown, while its restriction LinConj to *linear conjunctive grammars* equals the class of languages recognized by *real-time one-way one-dimensional* cellular automata. We prove two weakened versions of the open question $\text{Conj} \subseteq \text{RealTimeCA}$: 1) it is true for *unary* languages; 2) $\text{Conj} \subseteq \text{RealTime2OCA}$, i.e. any conjunctive language is recognized by a *real-time one-way two-dimensional* cellular automaton. Interestingly, we express the rules of a conjunctive grammar in two *Horn logics*, which *exactly characterize* the complexity classes RealTimeCA and RealTime2OCA .

Keywords: Computational complexity, Real-time, One-way/two-way communication, Grid-circuit, Unary language, Descriptive complexity, Existential second-order logic, Horn formula.

1 Introduction

For decades, logic has maintained close relationships with, on the one hand, computational models [27] and computational complexity [3], in particular through descriptive complexity [6, 14, 19, 9, 12, 2], and on the other hand with formal language theory and grammars [7, 19].

Conjunctive grammars versus logic: Okhotin [23] wrote that “context-free grammars may be thought of as a *logic* for inductive description of syntax in which the propositional connectives available... are restricted to *disjunction only*”. Thus, twenty years ago, the same author introduced *conjunctive grammars* [20] as an extension of context-free grammars by adding an explicit *conjunction* operation within the grammar rules.

As shown by Okhotin [20], conjunctive grammars – and more generally, Boolean grammars [21, 23] – inherit the parsing algorithms of the ordinary context-free grammars, without increasing their computational complexity. However, the expressive power of these grammars is largely unknown. The fact that the class Conj of languages generated by conjunctive grammars has many closure properties – it is trivially closed under reverse, concatenation, Kleene closure, disjunction and conjunction – suggests that this class has equivalent definitions in computational complexity and/or logic.

Conjunctive grammars versus real-time cellular automata: Note that the LinConj subclass of languages generated by *linear* conjunctive grammars was found to be equal to the Trellis class of languages recognized by *trellis* automata [22], or equivalently, *one-way real-time* cellular automata. Faced with this result, it is tempting to ask the following question: is the larger class Conj equal to the class RealTimeCA of languages recognized by *two-way* real-time cellular automata? Either answer to this question has strong consequences:

- If $\text{Conj} = \text{RealTimeCA}$ then each of the two classes will benefit from the closure properties of the other class; in particular, RealTimeCA would be closed under reverse, which was shown by [13] to imply $\text{RealTimeCA} = \text{LinearTimeCA}$, i.e. real-time is nothing but *linear time* for cellular automata, a surprising positive answer to a longstanding open question [5, 24, 26].
- If $\text{Conj} \neq \text{RealTimeCA}$ then $\text{Conj} \subsetneq \text{DSPACE}(n)$ or $\text{RealTimeCA} \subsetneq \text{DSPACE}(n)$: any of these strict inclusions would be a striking result.



© T. Grente, E. Grandjean;
licensed under Creative Commons License CC-BY 4.0
OpenAccess Series in Informatics

OASICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

46 Real-time is the minimal time of cellular automata (CA). Recall that `RealTimeCA` (resp.
 47 `Trellis`) is the class of languages recognized in real-time by *one-dimensional CA* with *two-way*
 48 (resp. *one-way*) communication and input word given in parallel. We know the strict inclusion
 49 $\text{Trellis} \subsetneq \text{RealTimeCA}$. The robustness of these classes is attested by their characteriz-
 50 ation by two sub-logics of ESO – the *existential second-order* logic, which characterizes
 51 NP – with *Horn formulas* as their first-order parts¹, and called respectively `pred-ESO-HORN`
 52 and `incl-ESO-HORN`, see [10, 11]. For short, we write $\text{RealTimeCA} = \text{pred-ESO-HORN}$ and
 53 $\text{Trellis} = \text{incl-ESO-HORN}$.

54 **Results of this paper:** This paper focuses on the relationships between the class of
 55 conjunctive languages and the real-time classes of cellular automata. Although we do not
 56 know the answer to the question $\text{Conj} =? \text{RealTimeCA}$ or even to the question of the inclusion
 57 $\text{Conj} \subseteq? \text{RealTimeCA}$, we prove two weakened versions of this inclusion:

- 58 1. $\text{Conj}_1 \subseteq \text{RealTimeCA}_1$: The inclusion holds when restricted to *unary* languages².
- 59 2. $\text{Conj} \subseteq \text{RealTime2OCA}$: The inclusion holds for real-time of *two-dimensional one-way*
 60 cellular automata (2-OCA). (We have $\text{RealTimeCA} \subseteq \text{RealTime2OCA}$.)

61 To grasp the scope of inclusion (1), it is important to note that unlike the subclass CFL_1
 62 of the unary languages of the class of context-free languages, which is reduced to regular
 63 languages, $\text{CFL}_1 = \text{Reg}_1$, the class Conj_1 was shown by Jez [15] to be much larger than Reg_1 .
 64 Understanding its precise expressiveness seems as difficult a problem to us as for `Conj`.

65 Our inclusion (2) improves the inclusion $\text{CFL} \subseteq \text{RealTime2SOCA}$, where `RealTime2SOCA`
 66 denotes the class of languages recognized by real-time *sequential* two-dimensional one-way
 67 cellular automata, proved by Terrier [25], who uses a result by King [16] and improves
 68 results by Kosaraju [18] and Chang et al. [4]. Terrier’s result derives transitively from (2):
 69 $\text{CFL} \subseteq \text{Conj} \subseteq \text{RealTime2OCA} \subseteq \text{RealTime2SOCA}$.

70 **Logic as a bridge from problems and grammars to real-time CAs:** Logic has been
 71 the basis of logic programming and database queries for decades, especially Horn logic through
 72 the Prolog and Datalog programming languages [1, 17, 9]. Likewise, the above-mentioned
 73 logical characterizations of real-time complexity classes of CAs, $\text{RealTimeCA} = \text{pred-ESO-HORN}$
 74 and $\text{Trellis} = \text{incl-ESO-HORN}$, have been used to easily show that several problems belong
 75 to the `RealTimeCA` or `Trellis` class by inductively expressing/programming the problems
 76 in the corresponding Horn logic, see [10, 11].

77 In this paper, the same logic programming method is adopted. We prove inclusion (1),
 78 $\text{Conj}_1 \subseteq \text{RealTimeCA}_1$, by expressing a unary language generated by a conjunctive grammar in
 79 the `pred-ESO-HORN` logic. Inclusion (1) follows, by the equality $\text{pred-ESO-HORN} = \text{RealTimeCA}$.
 80 Similarly, to prove inclusion (2), $\text{Conj} \subseteq \text{RealTime2OCA}$, we first design a logic denoted
 81 `incl-pred-ESO-HORN` so that $\text{incl-pred-ESO-HORN} = \text{RealTime2OCA}$. Then, we express any
 82 conjunctive language in this logic, proving that it belongs to `RealTime2OCA`, as claimed.

83 Thus, the heart of each proof consists in presenting a formula of a certain *Horn logic*,
 84 which *inductively* expresses how a word is generated by a conjunctive grammar: the Horn
 85 clauses of the formula *naturally* imitate the rules of the grammar.

86 **Our proof method and the paper structure:** After Section 2 gives some definitions,
 87 Sections 3 and 4 present inclusions (1) and (2) and their proofs with a common plan:
 88 Subsection 3.1 (resp. 4.1) expresses the inductive generating process of a conjunctive
 89 grammar, assumed in binary (Chomsky) normal form in the logic `pred-ESO-HORN` (resp.

¹ The class ESO-HORN of languages defined by existential second-order formulas with Horn formulas as their first-order parts is exactly PTIME, see [8].

² The subclass of the unary languages of a class of languages \mathcal{C} is denoted \mathcal{C}_1 .

incl-pred-ESO-HORN). Subsection 3.2 (resp. 4.2) shows that any formula of this logic can be normalized into a formula which mimics the computation of a two-dimensional (resp. three-dimensional) *grid-circuit* called **Grid** (resp. **Cube**); Subsection 3.3 (resp. 4.3) translates the grid-circuit into a real-time one-dimensional CA (resp. two-dimensional OCA). Note that we prove the equivalence of our logics with grid-circuits and CA real-time³: $\text{pred-ESO-HORN} = \text{Grid} = \text{RealTimeCA}$ and $\text{incl-pred-ESO-HORN} = \text{Cube} = \text{RealTime2OCA}$. Section 5 gives a conclusion with a diagram of the known relations between the **Conj** class and the CA complexity classes studied here, for the general case and for the unary case.

2 Preliminaries

2.1 Conjunctive grammars and their binary normal form

Conjunctive grammars extend context-free grammars with a conjunction operation.

► **Definition 1.** A conjunctive grammar is a tuple $G = (\Sigma, N, P, S)$ where Σ is the finite set of terminal symbols, N is the finite set of nonterminal symbols, $S \in N$ is the initial symbol, and P is the finite set of rules, each of the form $A \rightarrow \alpha_1 \& \dots \& \alpha_m$, for $m \geq 1$ and $\alpha_i \in (\Sigma \cup N)^+$. The set of words, $L(A)$, generated by any $A \in N$ is defined by induction: if the rules for A are $A \rightarrow \alpha_1^1 \& \dots \& \alpha_{m_1}^1 \mid \dots \mid \alpha_1^k \& \dots \& \alpha_{m_k}^k$, then $L(A) := \bigcup_{i=1}^k \bigcap_{j=1}^{m_i} L(\alpha_j^i)$. The language generated by the grammar G is $L(S)$.

We will mainly use the binary normal form of conjunctive grammars, which extends the Chomsky normal form of context-free grammars. Each conjunctive grammar can be rewritten in an equivalent binary normal form [20, 23].

► **Definition 2** (Binary normal form [20]). A conjunctive grammar $G = (\Sigma, N, P, S)$ is in binary normal form if each rule in P has one of the two following forms:

- a long rule: $A \rightarrow B_1 C_1 \& \dots \& B_m C_m$ ($m \geq 1, B_i, C_j \in N$);
- a short rule: $A \rightarrow a$ ($a \in \Sigma$).

2.2 Elements of logic

The underlying structure we will use to encode an input word $w = w_1 \dots w_n$ on its index interval $[1, n] = \{1, \dots, n\}$ uses the *successor* and *predecessor* functions and the monadic predicates **min** and **max** as its *only* arithmetic functions/predicates:

► **Definition 3** (structure encoding a word). Each nonempty word $w = w_1 \dots w_n \in \Sigma^n$ on a fixed finite alphabet Σ is represented by the first-order structure $\langle w \rangle := ([1, n]; (Q_s)_{s \in \Sigma}, \text{min}, \text{max}, \text{suc}, \text{pred})$ of domain $[1, n]$, monadic predicates Q_s , $s \in \Sigma$, **min** and **max** such that $Q_s(i) \iff w_i = s$, **min**(i) $\iff i = 1$, and **max**(i) $\iff i = n$, and unary functions **suc** and **pred** such that **suc**(i) = $i + 1$ for $i < n$ and **suc**(n) = n , **pred**(i) = $i - 1$ for $i > 1$ and **pred**(1) = 1 . Let \mathcal{S}_Σ denote the signature $\{(Q_s)_{s \in \Sigma}, \text{min}, \text{max}, \text{suc}, \text{pred}\}$ of the structure $\langle w \rangle$.

► **Notation 1.** Let $x + k$ and $x - k$ abbreviate the terms **suc** ^{k} (x) and **pred** ^{k} (x), for a fixed integer $k \geq 0$. We will also use the intuitive abbreviations $x = 1$, $x = n$ and $x > k$, for a fixed integer $k \geq 1$, in place of the formulas **min**(x), **max**(x) and $\neg \text{min}(x - (k - 1))$, respectively.

³ We have chosen to give here a simplified proof of the logical characterization $\text{pred-ESO-HORN} = \text{Grid} = \text{RealTimeCA}$ already proved in [10] so that this paper is self-content, but above all because our proof of the similar result $\text{incl-pred-ESO-HORN} = \text{Cube} = \text{RealTime2OCA}$ is an extension of it.

128 **2.3 Cellular automata and real-time**

129 ▶ **Definition 4** (1-CA and 2-OCA). A d -dimensional cellular automaton (CA) is a triple
 130 $(\mathbf{S}, \mathcal{N}, \mathbf{f})$ where \mathbf{S} is the finite set of states, $\mathcal{N} \subset \mathbb{Z}^d$ is the neighborhood, and $\mathbf{f} : \mathbf{S}^{|\mathcal{N}|} \rightarrow \mathbf{S}$
 131 is the transition function. We are interested in the following two special cases:

132 ■ 1-CA: It is a one-dimensional two-way cellular automaton $(\mathbf{S}, \{-1, 0, 1\}, \mathbf{f})$, for which the
 133 state $\langle c, t \rangle$ of any cell c at a time $t > 1$ is updated in this way:
 134 $\langle c, t \rangle = \mathbf{f}(\langle c-1, t-1 \rangle, \langle c, t-1 \rangle, \langle c+1, t-1 \rangle)$.

135 ■ 2-OCA: It is a two-dimensional one-way cellular automaton $(\mathbf{S}, \{(0, 0), (-1, 0), (0, -1)\}, \mathbf{f})$
 136 for which the state $\langle c_1, c_2, t \rangle$ of any cell (c_1, c_2) at a time $t > 1$ is updated in this way:
 137 $\langle c_1, c_2, t \rangle = \mathbf{f}(\langle c_1, c_2, t-1 \rangle, \langle c_1-1, c_2, t-1 \rangle, \langle c_1, c_2-1, t-1 \rangle)$.

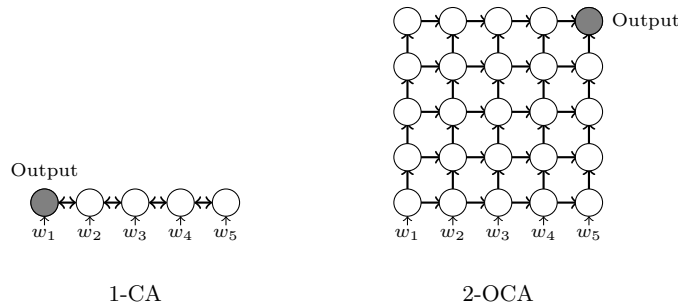
138 ▶ **Definition 5** (permanent and quiescent states). In a CA, a state \sharp is permanent if a cell
 139 in state \sharp remains in this state forever. A state λ of a CA is quiescent if a cell in state λ
 140 remains in this state as long as the states of its neighborhood cells are quiescent or permanent.

141 ▶ **Definition 6** (CA as a word acceptor). A cellular automaton $(\mathbf{S}, \mathcal{N}, \mathbf{f})$ with an input
 142 alphabet $\Sigma \subset \mathbf{S}$, a permanent state \sharp , a quiescent state λ , and a set of accepting states
 143 $\mathbf{S}_{\text{acc}} \subset \mathbf{S}$ acts as a word acceptor if it operates on an input word $w \in \Sigma^+$ in respecting the
 144 following conditions (see Figure 1).

145 **Input.** For a 1-CA, the i -th symbol of the input $w = w_1 \dots w_n$ is given to the cell i at the
 146 initial time 1: $\langle i, 1 \rangle = w_i$. All other cells are in the permanent state \sharp . For a 2-OCA, the
 147 i -th symbol of the input is given to the cell $(i, 1)$ at time 1: $\langle i, 1, 1 \rangle = w_i$. At time 1, the cells
 148 $(c_1, c_2) \in [1, n] \times [2, n]$ are in the quiescent state λ , all other cells are in the permanent state \sharp .

149 **Output.** One specific cell called output cell gives the output, “accept” or “reject”, of the
 150 computation. For a 1-CA, the output cell is the cell 1. For a 2-OCA, the output cell is (n, n) .

151 **Acceptance.** An input word is accepted by a 1-CA (resp. 2-CA) at time t if the output cell
 152 enters an accepting state at time t .



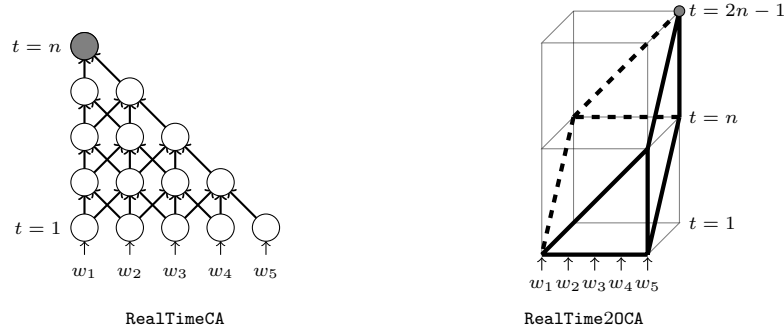
■ **Figure 1** Input and output of a CA acting as a word acceptor

153 ▶ **Definition 7** (RealTimeCA, RealTime2OCA). A word is accepted in real-time by a 1-CA
 154 (resp. 2-OCA) if the word is accepted in minimal time for the output cell 1 (resp. (n, n)) to
 155 receive each of its letters. A language is recognized in real-time by a CA if it is the set of
 156 words that it accepts in real-time. The class RealTimeCA (resp. RealTime2OCA) is the class
 157 of languages recognized in real-time by a 1-CA (resp. 2-OCA).

158 **3 Real-time recognition of a unary conjunctive language**

159 In this section, we prove our first main result:

160 ▶ **Theorem 8.** $\text{Conj}_1 \subseteq \text{RealTimeCA}_1$.



■ **Figure 2** Space-time diagrams of RealTimeCA and RealTime20CA

3.1 Expressing inductively a unary conjunctive language in logic

The generating process of a unary conjunctive language is naturally expressed in the logic **pred-ESO-HORN**, an inductive Horn logic whose only function is the predecessor function.

► **Definition 9** (**pred-ESO-HORN**). A formula of **pred-ESO-HORN** is a formula $\Phi := \exists \mathbf{R} \forall x \forall y \psi(x, y)$ where \mathbf{R} is a finite set of binary predicates and ψ is a conjunction of Horn clauses, of signature $\mathcal{S}_\Sigma \cup \mathbf{R}$, and of one the three following forms:

- an input clause: $\min(x) \wedge (\neg) \min(y) \wedge Q_s(y) \rightarrow R(x, y)$ with $s \in \Sigma$ and $R \in \mathbf{R}$;
- a computation clause: $\delta_1 \wedge \dots \wedge \delta_r \rightarrow R(x, y)$ with $R \in \mathbf{R}$ and where each hypothesis δ_i is an atom $S(x, y)$ or a conjunction $S(x - a, y - b) \wedge x > a \wedge y > b$, with $S \in \mathbf{R}$ and $a, b \geq 0$ two integers such that $a + b > 0$;
- a contradiction clause: $\max(x) \wedge \max(y) \wedge R(x, y) \rightarrow \perp$ with $R \in \mathbf{R}$.

We denote by **pred-ESO-HORN** the class of languages defined by a formula of **pred-ESO-HORN**.

► **Remark 10.** We will freely use equalities (resp. inequalities) $x = a$ and $y = b$ (resp. $x > a$, $y > b$), for constants a, b , in our formulas since they can be easily defined in **pred-ESO-HORN**. For example, the binary predicate $R^{x>2}$ of intuitive meaning $R^{x>2}(x, y) \iff x > 2$ is defined inductively by the following clauses where $R^{x=a}(x, y)$ means $x = a$:

- $\min(x) \rightarrow R^{x=1}(x, y)$; $x > 1 \wedge R^{x=1}(x - 1, y) \rightarrow R^{x=2}(x, y)$;
- $x > 1 \wedge R^{x=2}(x - 1, y) \rightarrow R^{x>2}(x, y)$; $x > 1 \wedge R^{x>2}(x - 1, y) \rightarrow R^{x>2}(x, y)$.

Also, some other arithmetic predicates easily defined in **pred-ESO-HORN** will be used. For example, $y = 2x$ can be replaced by the atom $R^{y=2x}(x, y)$, where $R^{y=2x}$ is defined by the following two clauses using the predicates $R^{x=1}$, $R^{y=2}$, $R^{x>1}$ and $R^{y>2}$:

- $x = 1 \wedge y = 2 \rightarrow R^{y=2x}(x, y)$; $x > 1 \wedge y > 2 \wedge R^{y=2x}(x - 1, y - 2) \rightarrow R^{y=2x}(x, y)$.

► **Notation 2.** More generally, let $R^{\rho(x,y)}$ denote a binary predicate whose meaning is $R^{\rho(x,y)}(x, y) \iff \rho(x, y)$, for a property or a formula $\rho(x, y)$. We will also use a set of binary arithmetic predicates denoted by $\mathbf{R}_{\text{arith}}$, which consists of $R^{x=y}$, $R^{y=2x}$ and $R^{\rho(x,y)}$, for $\rho(x, y) := x \geq \lceil \frac{y}{2} \rceil$, and the predicates used to define them in **pred-ESO-HORN**.

Let us prove that for every unary conjunctive languages, their complements can be defined in **pred-ESO-HORN**₁.

► **Lemma 11.** For each language $L \subseteq a^+$, if $L \in \text{Conj}_1$ then $a^+ \setminus L \in \text{pred-ESO-HORN}$.

Proof. Let $G = (\{a\}, N, P, S)$ be a conjunctive grammar in binary normal form which generates L . For each $A \in N$ and each unary word a^y , we have, according to the length y , the following equivalences which will be the basis of our induction:

XX:6 Conjunctive grammars, cellular automata and logic

- 193 ■ if $y = 1$, then $a^y \in L(A) \iff$ the short rule $A \rightarrow a$ belongs to P ;
 194 ■ if $y > 1$, then $a^y \in L(A) \iff$ there is a long rule $A \rightarrow B_1C_1 \& \dots \& B_mC_m$
 195 in P such that, for each $i \in \{1, \dots, m\}$, there exists $x \geq \lceil \frac{y}{2} \rceil$ such that
 196 either $a^x \in L(B_i)$ and $a^{y-x} \in L(C_i)$, or $a^{y-x} \in L(B_i)$ and $a^x \in L(C_i)$.

197 We want to construct a first-order formula $\forall x \forall y \psi_G(x, y)$ of signature $\mathcal{S}_\Sigma \cup \mathbf{R}$, for $\Sigma := \{a\}$
 198 and the set of binary predicates $\mathbf{R} := \{\text{Maj}_A, \text{Min}_A \mid A \in N\} \cup \{\text{Sum}_{BC} \mid B, C \in N\} \cup \mathbf{R}_{\text{arith}}$
 199 so that the formula $\Phi_G := \exists \mathbf{R} \forall x \forall y \psi_G$ belongs to **pred-ESO-HORN** and defines the language
 200 $a^+ \setminus L$. The intuitive meanings of the predicates $\text{Maj}_A, \text{Min}_A$ and Sum_{BC} are as follows:

- 201 ■ $\text{Maj}_A(x, y) \iff \lceil \frac{y}{2} \rceil \leq x \leq y$ and $a^x \in L(A)$;
 202 ■ $\text{Min}_A(x, y) \iff \lceil \frac{y}{2} \rceil \leq x < y$ and $a^{y-x} \in L(A)$;
 203 ■ $\text{Sum}_{BC}(x, y) \iff$ there is some x' with $\lceil \frac{y}{2} \rceil \leq x' \leq x$ such that
 204 either $a^{x'} \in L(B)$ and $a^{y-x'} \in L(C)$, or $a^{y-x'} \in L(B)$ and $a^{x'} \in L(C)$.

205 Note that for $x = y$, the above equivalence for Maj_A implies $\text{Maj}_A(x, y) \iff a^y \in L(A)$.

206 Let us give and justify a list of Horn clauses whose conjunction ψ'_G defines the predicates
 207 $\text{Maj}_A, \text{Min}_A$ and Sum_{BC} , using the arithmetic predicates of $\mathbf{R}_{\text{arith}}$ (see Notation 2 and
 208 Remark 10), namely $R^{x=y}, R^{y=2x}$ and $R^{\rho(x,y)}$, for $\rho(x, y) := x \geq \lceil \frac{y}{2} \rceil$.

209 **Short rules.** Each rule $A \rightarrow a$ of P is expressed by the input clause:

- 210 ■ $\text{min}(x) \wedge \text{min}(y) \wedge Q_a(y) \rightarrow \text{Maj}_A(x, y)$.

211 **Induction on the length y .** If we have for $y > 1$ the inequalities $\lceil \frac{y-1}{2} \rceil \leq x \leq y-1$ and
 212 $x \geq \lceil \frac{y}{2} \rceil$ then $\lceil \frac{y}{2} \rceil \leq x \leq y$. This justifies the clause:

- 213 ■ $y > 1 \wedge \text{Maj}_A(x, y-1) \wedge x \geq \lceil \frac{y}{2} \rceil \rightarrow \text{Maj}_A(x, y)$ for all $A \in N$.

214 For $y > 1$ and $y = 2x$, we have $a^x = a^{y-x}$ and $\lceil \frac{y}{2} \rceil \leq x < y$. This justifies the clause:

- 215 ■ $y > 1 \wedge \text{Maj}_A(x, y-1) \wedge y = 2x \rightarrow \text{Min}_A(x, y)$ for all $A \in N$.

216 If for $x, y > 1$ we have the inequalities $\lceil \frac{y-1}{2} \rceil \leq x-1 < y$, then $\lceil \frac{y}{2} \rceil \leq x < y$. Moreover,
 217 $a^{(y-1)-(x-1)} = a^{y-x}$. This justifies the clause:

- 218 ■ $x > 1 \wedge y > 1 \wedge \text{Min}_A(x-1, y-1) \rightarrow \text{Min}_A(x, y)$ for all $A \in N$.

219 **Concatenation.** For all $B, C \in N$, it is clear that the concatenation predicate Sum_{BC} is
 220 defined inductively by the following three clauses:

- 221 ■ *initialization:* $\text{Maj}_B(x, y) \wedge \text{Min}_C(x, y) \rightarrow \text{Sum}_{BC}(x, y)$; $\text{Min}_B(x, y) \wedge \text{Maj}_C(x, y) \rightarrow \text{Sum}_{BC}(x, y)$;
 222 ■ *induction:* $\neg \text{min}(x) \wedge \text{Sum}_{BC}(x-1, y) \rightarrow \text{Sum}_{BC}(x, y)$.

223 **Long rules.** Each rule $A \rightarrow B_1C_1 \& \dots \& B_mC_m$ of P is expressed by the clause:

- 224 ■ $x = y \wedge \text{Sum}_{B_1C_1}(x, y) \wedge \dots \wedge \text{Sum}_{B_mC_m}(x, y) \rightarrow \text{Maj}_A(x, y)$.

225 Thus, the formula $\forall x \forall y \psi'_G$ where ψ'_G is the conjunction of the above clauses defines the
 226 predicates $\text{Maj}_A, \text{Min}_A$, and Sum_{BC} .

227 **Definition of $a^+ \setminus L$.** We have the equivalence $\text{Maj}_S(n, n) \iff a^n \in L(S) \iff a^n \in L$.

228 Therefore, the following contradiction clause expresses $a^n \notin L$:

- 229 ■ $\gamma_S := \text{max}(x) \wedge \text{max}(y) \wedge \text{Maj}_S(x, y) \rightarrow \perp$.

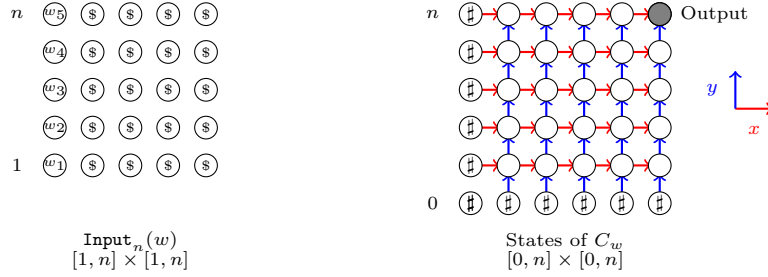
230 Finally, observe that the formula $\Phi_G := \exists \mathbf{R} \forall x \forall y \psi_G$ where ψ_G is $\gamma_{\text{arith}} \wedge \psi'_G \wedge \gamma_S$ and
 231 γ_{arith} is the conjunction of clauses that defines the arithmetic predicates of $\mathbf{R}_{\text{arith}}$, belongs
 232 to **pred-ESO-HORN**. Since we have $\langle a^n \rangle \models \Phi_G \iff a^n \notin L$, as justified above, then the
 233 language $a^+ \setminus L$ belongs to **pred-ESO-HORN**, as claimed. ◀

234 3.2 Equivalence of logic with grid-circuits

235 We introduce the *grid-circuit* as an intermediate object between our logic and the real-time
236 cellular automaton: see Figure 3.

- 237 ► **Definition 12.** A grid-circuit is a tuple $C := (\Sigma, (\text{Input}_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{\text{acc}}, \mathbf{g})$ where
- 238 ■ Σ is the input alphabet and $(\text{Input}_n)_{n>0}$ is the family of input functions
 - 239 $\text{Input}_n : \Sigma^n \times [1, n]^2 \rightarrow \Sigma \cup \{\$\}$ such that, for $w = w_1 \dots w_n \in \Sigma^n$,
 - 240 $\text{Input}_n(w, x, y) = w_y$ if $x = 1$ and $\text{Input}_n(w, x, y) = \$$ otherwise,
 - 241 ■ $\mathbf{Q} \cup \{\#\}$ is the finite set of states and $\mathbf{Q}_{\text{acc}} \subseteq \mathbf{Q}$ is the subset of accepting states,
 - 242 ■ $\mathbf{g} : (\mathbf{Q} \cup \{\#\})^2 \times (\Sigma \cup \{\$\}) \rightarrow \mathbf{Q}$ is the transition function.

- 243 ► **Definition 13** (computation of a grid-circuit). The computation C_w of a grid-circuit
244 $C := (\Sigma, (\text{Input}_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{\text{acc}}, \mathbf{g})$ on a $w = w_1 \dots w_n \in \Sigma^n$ is a regular grid of $(n+1)^2$ sites
245 $(x, y) \in [0, n]^2$, each in a state $\langle x, y \rangle \in \mathbf{Q} \cup \{\#\}$ computed inductively:
- 246 ■ each site in $\{0\} \times [0, n]$ or $[0, n] \times \{0\}$ is in the particular state $\#$;
 - 247 ■ the state of each site $(x, y) \in [1, n]^2$ is $\langle x, y \rangle = \mathbf{g}(\langle x, y-1 \rangle, \langle x-1, y \rangle, \text{Input}_n(w, x, y))$.



■ **Figure 3** The grid-circuit

248 A word $w = w_1 \dots w_n \in \Sigma^n$ is *accepted* by the grid-circuit C if the output state $\langle n, n \rangle$ of C_w
249 belongs to \mathbf{Q}_{acc} . The language *recognized* by C is the set of words it accepts. We denote by
250 **Grid** the class of languages recognized by a grid-circuit.

251 Actually, our predecessor Horn logic is equivalent to grid-circuits.

252 ► **Lemma 14.** [10] $\text{pred-ESO-HORN} = \text{Grid}$.

253 **Proof.** In some sense, a grid-circuit is the “normalized form” of a formula of **pred-ESO-HORN**.
254 So, the inclusion $\text{Grid} \subseteq \text{pred-ESO-HORN}$ is proved in a straightforward way.

255 The first step of the proof of the converse inclusion $\text{pred-ESO-HORN} \subseteq \text{Grid}$ is to show
256 that every formula $\Phi := \exists \mathbf{R} \forall x \forall y \psi(x, y)$ in **pred-ESO-HORN** is equivalent to a formula
257 $\Phi' \in \text{pred-ESO-HORN}$ in which the only hypotheses of computation clauses are atoms $S(x, y)$
258 and conjunctions $S(x-1, y) \wedge x > 1$ and $S(x, y-1) \wedge y > 1$.

259 **Elimination of atoms $R(x-a, y-b)$ for $a+b > 1$:** The idea is to introduce new “shift”
260 predicates $R^{x-a', y-b'}$ for fixed integers $a', b' > 0$ with the intuitive meaning:

$$261 R^{x-a', y-b'}(x, y) \iff R(x-a', y-b') \wedge x > a' \wedge y > b'.$$

262 Let us explain the method by an example. Assume we have in ψ the Horn clause

263 (1) $x > 3 \wedge y > 2 \wedge S(x-3, y-2) \rightarrow T(x, y)$. This clause is replaced by the clause

264 (2) $S^{x-2, y-2}(x-1, y) \wedge x > 1 \rightarrow T(x, y)$

265 for which the predicates S^{x-1} , S^{x-2} , $S^{x-2, y-1}$ and $S^{x-2, y-2}$ are defined by the respect-

266 ive clauses: $x > 1 \wedge S(x-1, y) \rightarrow S^{x-1}(x, y)$, $x > 1 \wedge S^{x-1}(x-1, y) \rightarrow S^{x-2}(x, y)$,

267 $y > 1 \wedge S^{x-2}(x, y-1) \rightarrow S^{x-2, y-1}(x, y)$, and $y > 1 \wedge S^{x-2, y-1}(x, y-1) \rightarrow S^{x-2, y-2}(x, y)$,

XX:8 Conjunctive grammars, cellular automata and logic

268 which imply together the clause $x > 2 \wedge y > 2 \wedge S(x-2, y-2) \rightarrow S^{x-2, y-2}(x, y)$ and then
 269 also $x > 3 \wedge y > 2 \wedge S(x-3, y-2) \rightarrow S^{x-2, y-2}(x-1, y)$.

270 It is clear that the formula $\Phi := \exists \mathbf{R} \forall x \forall y \psi$ is equivalent to the formula $\Phi' := \exists \mathbf{R}' \forall x \forall y \psi'$
 271 where $\mathbf{R}' := \mathbf{R} \cup \{S^{x-1}, S^{x-2}, S^{x-2, y-1}, S^{x-2, y-2}\}$ and ψ' is the conjunction $\psi_{\text{replace}} \wedge \psi_{\text{def}}$,
 272 where ψ_{replace} is the formula ψ in which clause (1) is replaced by clause (2), and ψ_{def} is the
 273 conjunction of the above clauses defining the new predicates of \mathbf{R}' .

274 Thus, any formula $\Phi \in \text{pred-ESO-HORN}$ is equivalent to a formula $\Phi' \in \text{pred-ESO-HORN}$
 275 whose computation clauses only contain hypotheses of the following three forms:
 276 $R(x-1, y) \wedge x > 1$; $R(x, y-1) \wedge y > 1$; $R(x, y)$. The next step is to eliminate these $R(x, y)$.

Elimination of hypotheses $R(x, y)$ (sketch of proof): The first idea is to group together
 in each computation clause the hypothesis atoms of the form $R(x, y)$ and the conclusion of
 the clause. As a result, the formula can be rewritten in the form

$$\Phi := \exists \mathbf{R} \forall x \forall y \left[\bigwedge_i C_i(x, y) \wedge \bigwedge_{i \in [1, k]} (\alpha_i(x, y) \rightarrow \theta_i(x, y)) \right]$$

277 where the C_i 's are the input clauses and the contradiction clauses, and each computation
 278 clause is written in the form $\alpha_i(x, y) \rightarrow \theta_i(x, y)$, where $\alpha_i(x, y)$ is a conjunction of formulas
 279 of the only forms $R(x-1, y) \wedge x > 1$, $R(x, y-1) \wedge y > 1$, and $\theta_i(x, y)$ is a Horn clause in
 280 which *all* atoms are of the form $R(x, y)$.

281 The second idea is to “solve” the Horn clauses θ_i according to the input clauses and *all*
 282 *the possible* conjunctions of hypotheses α_i that may be true. Notice the two following facts:
 283 the hypotheses of the input clauses are input literals and the conjuncts of the α_i 's are of the
 284 only forms $R(x-1, y) \wedge x > 1$, $R(x, y-1) \wedge y > 1$. So, we can prove by induction on the sum
 285 $x + y$ that the obtained formula Φ' in which no atom $R(x, y)$ appears as a clause hypothesis,
 286 is equivalent to the above formula Φ . The complete proof is given in Appendix A.

287 **Transformation of the formula into a grid-circuit:** Let $\mathbf{R} = \{R_1, \dots, R_m\}$ denote
 288 the set of binary predicates of the formula. By a separation into cases of input clauses
 289 and computation clauses, it is easy to transform the formula into an equivalent formula
 290 $\Phi := \exists \mathbf{R} \forall x \forall y \psi$ where ψ is a conjunction of clauses of the following forms (a-e), in which
 291 $s \in \Sigma$, $j \in [1, m]$, and A, B are (possibly empty) subsets of $[1, m]$:

- 292 (a) $x = 1 \wedge y = 1 \wedge Q_s(y) \rightarrow R_j(x, y)$;
- 293 (b) $x = 1 \wedge y > 1 \wedge Q_s(y) \wedge \bigwedge_{i \in A} R_i(x, y-1) \rightarrow R_j(x, y)$;
- 294 (c) $x > 1 \wedge y = 1 \wedge \bigwedge_{i \in A} R_i(x-1, y) \rightarrow R_j(x, y)$;
- 295 (d) $x > 1 \wedge y > 1 \wedge \bigwedge_{i \in A} R_i(x-1, y) \wedge \bigwedge_{i \in B} R_i(x, y-1) \rightarrow R_j(x, y)$;
- 296 (e) $x = n \wedge y = n \wedge R_j(x, y) \rightarrow \perp$.

297 Now, transform this formula into a grid-circuit $C := (\Sigma, (\text{Input}_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{\text{acc}}, \mathbf{g})$. The
 298 idea is that the state of a site $(x, y) \in [1, n]^2$ is the set of predicates R_i such that $R_i(x, y)$ is
 299 true. Let \mathbf{Q} be the power set of the set of \mathbf{R} indices: $\mathbf{Q} := \mathcal{P}([1, m])$. There are four types of
 300 transition (a-d) which mimic the clauses (a-d) above. These are, for $s \in \Sigma$ and $q, q' \in \mathbf{Q}$:

- 301 (a) $\mathbf{g}(\#, \#, s) = \{j \in [1, m] \mid \text{there is a clause (a) with } Q_s, \text{ and conclusion } R_j(x, y)\}$;
- 302 (b) $\mathbf{g}(q, \#, s) = \{j \in [1, m] \mid \text{there is a clause (b) with } Q_s, \text{ and } A \subseteq q, \text{ and conclusion } R_j(x, y)\}$;
- 303 (c) $\mathbf{g}(\#, q, \$) = \{j \in [1, m] \mid \text{there is a clause (c) with } A \subseteq q, \text{ and conclusion } R_j(x, y)\}$;
- 304 (d) $\mathbf{g}(q, q', \$) = \{j \in [1, m] \mid \exists \text{ a clause (d) with } A \subseteq q, B \subseteq q', \text{ and conclusion } R_j(x, y)\}$.

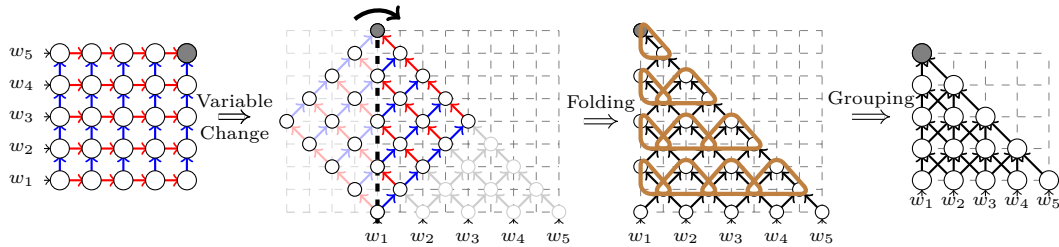
305 Of course, the set of accepting states of C is determined by the contradiction clauses (e):
 306 $\mathbf{Q}_{\text{acc}} := \{q \in \mathbf{Q} \mid q \text{ contains no } j \text{ such that } R_j \text{ occurs in a clause (e)}\}$.

307 We can easily check the equivalence, for each $w \in \Sigma^+$: $\langle w \rangle \models \Phi \iff C \text{ accepts } w$.
 308 Therefore, the inclusion $\text{pred-ESO-HORN} \subseteq \text{Grid}$ is proved. \blacktriangleleft

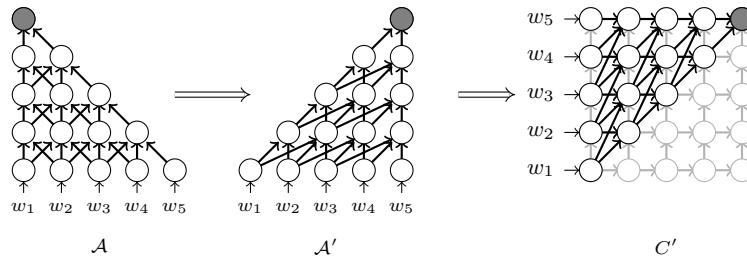
3.3 Grid-circuits are equivalent to real-time 1-CA

► **Lemma 15.** [10] $\text{Grid} = \text{RealTimeCA}$.

Proof. Figure 4 shows how Grid is simulated on RealTimeCA and Figure 5 shows how RealTimeCA is simulated on Grid . The proof is detailed in Appendix B. ◀



■ **Figure 4** Simulation of Grid on RealTimeCA



■ **Figure 5** Simulation of RealTimeCA on the grid-circuit

312

313 **Proof of Theorem 8.** Lemmas 14 and 15 give us the following equalities of classes:
 314 $\text{pred-ESO-HORN} = \text{Grid} = \text{RealTimeCA}$. These equalities trivially hold when restricted
 315 to unary languages: $\text{pred-ESO-HORN}_1 = \text{Grid}_1 = \text{RealTimeCA}_1$.

316 From the fact that the class RealTimeCA_1 is closed under complement and from Lemma 11,
 317 we deduce $\text{Conj}_1 \subseteq \text{pred-ESO-HORN}_1 = \text{Grid}_1 = \text{RealTimeCA}_1$. ◀

4 Real-time recognition of a conjunctive language: the general case

319 Our second main result strengthens the inclusion $\text{CFL} \subseteq \text{RealTime2SOCA}$ of Terrier⁴ [25]:

320 ► **Theorem 16.** $\text{Conj} \subseteq \text{RealTime2OCA}$.

4.1 Expressing a conjunctive language in logic: the general case

322 The generating process of a conjunctive language is naturally expressed in the Horn logic
 323 $\text{incl-pred-ESO-HORN}$. This is a hybrid logic with three first-order variables x, y, z , whose
 324 name means that it makes inductions on the variable interval $[x, y]$, by *inclusion*, and on the
 325 individual variable z , by *predecessor*.

⁴ Recall that RealTime2SOCA is the class of languages recognized by *sequential* one-way two-dimensional cellular automata in real-time: this is the minimal time, $3n - 1$, for the output cell (n, n) to receive the n letters of the input word, communicated sequentially by the input cell $(1, 1)$.

XX:10 Conjunctive grammars, cellular automata and logic

326 ► **Definition 17** (incl-pred-ESO-HORN). A formula of incl-pred-ESO-HORN is a formula
 327 $\Phi := \exists \mathbf{R} \forall x \forall y \forall z \psi(x, y, z)$ where \mathbf{R} is a finite set of ternary predicates, and ψ is a conjunction
 328 of Horn clauses, of signature $\mathcal{S}_\Sigma \cup \mathbf{R} \cup \{=, \leq\}$, and of the three following forms:

- 329 ■ an input clause: $x = y \wedge \min(z) \wedge Q_s(x) \rightarrow R(x, y, z)$ with $s \in \Sigma$ and $R \in \mathbf{R}$;
- 330 ■ a computation clause: $\delta_1 \wedge \dots \wedge \delta_r \rightarrow R(x, y, z)$ with $R \in \mathbf{R}$ and where each hypothesis δ_i
 331 is an atom $S(x, y, z)$ or a conjunction $S(x + a, y - b, z - c) \wedge x + a \leq y - b \wedge z > c$ with
 332 $S \in \mathbf{R}$ and $a, b, c \geq 0$ three integers such that $a + b + c > 0$;
- 333 ■ a contradiction clause: $\min(x) \wedge \max(y) \wedge \max(z) \wedge R(x, y, z) \rightarrow \perp$ with $R \in \mathbf{R}$.

334 We denote by incl-pred-ESO-HORN the class of languages defined by a formula of incl-pred-ESO-HORN.

335 ► **Lemma 18.** For each language $L \subseteq \Sigma^+$, if $L \in \text{Conj}$, then $\Sigma^+ \setminus L \in \text{incl-pred-ESO-HORN}$.

336 **Proof.** The proof is a variation (an extension) of the proof of the same result, Lemma 11, in
 337 the unary case. This is why we insist on the differences. Let $G = (\Sigma, N, P, S)$ be a conjunctive
 338 grammar in binary normal form which generates L and let w be a word $w = w_1 \dots w_n \in \Sigma^+$.
 339 For each $A \in N$ and each factor $w_{x,y} := w_x \dots w_y$, we have, according to the length $y - x + 1$
 340 of $w_{x,y}$, the following equivalences which will be the basis of our induction:

- 341 ■ if $x = y$, then $w_{x,y} \in L(A) \iff$ the short rule $A \rightarrow w_x$ belongs to P ;
- 342 ■ if $x < y$, then $w_{x,y} \in L(A) \iff$ there is a long rule $A \rightarrow B_1 C_1 \& \dots \& B_m C_m$
 343 in P such that, for each $i \in \{1, \dots, m\}$, there exists $z \geq \lceil (y - x + 1)/2 \rceil$ such that
 344 either $w_{x, x+z-1} \in L(B_i)$ and $w_{x+z, y} \in L(C_i)$, or $w_{x, y-z} \in L(B_i)$ and $w_{y-z+1, y} \in L(C_i)$.

345 Thus, a double induction is performed, on the index interval $[x, y]$ of a factor $w_{x,y}$ and
 346 on the maximal z among the lengths of the two sub-factors u, v of the m decompositions
 347 $w_{x,y} = uv$, $u \in L(B_i)$, $v \in L(C_i)$, for a long rule. This is naturally expressed in the logic
 348 incl-pred-ESO-HORN.

349 We want to construct a first-order formula $\forall x \forall y \forall z \psi_G$ of signature $\mathcal{S}_\Sigma \cup \mathbf{R} \cup \{=, \leq\}$,
 350 for the set of ternary predicates $\mathbf{R} := \{\text{Pref}_A^{\text{Maj}}, \text{Pref}_A^{\text{Min}}, \text{Suff}_A^{\text{Maj}}, \text{Suff}_A^{\text{Min}} \mid A \in N\} \cup$
 351 $\{\text{Concat}_{BC} \mid B, C \in N\} \cup \mathbf{R}_{\text{arith}}$, so that the formula $\Phi_G := \exists \mathbf{R} \forall x \forall y \forall z \psi_G$ belongs to
 352 incl-pred-ESO-HORN and defines the language $\Sigma^+ \setminus L$. The intuitive meanings of the predic-
 353 ates $\text{Pref}_A^{\text{Maj}}, \text{Pref}_A^{\text{Min}}, \text{Suff}_A^{\text{Maj}}, \text{Suff}_A^{\text{Min}}$ and Concat_{BC} are as follows:

- 354 ■ $\text{Pref}_A^{\text{Maj}}(x, y, z) \iff \lceil \frac{y-x+1}{2} \rceil \leq z \leq y - x + 1$ and $w_{x, x+z-1} \in L(A)$;
- 355 ■ $\text{Pref}_A^{\text{Min}}(x, y, z) \iff \lceil \frac{y-x+1}{2} \rceil \leq z \leq y - x$ and $w_{x, y-z} \in L(A)$;
- 356 ■ $\text{Suff}_A^{\text{Maj}}(x, y, z) \iff \lceil \frac{y-x+1}{2} \rceil \leq z \leq y - x + 1$ and $w_{y-z+1, y} \in L(A)$;
- 357 ■ $\text{Suff}_A^{\text{Min}}(x, y, z) \iff \lceil \frac{y-x+1}{2} \rceil \leq z \leq y - x$ and $w_{x+z, y} \in L(A)$;
- 358 ■ $\text{Concat}_{BC}(x, y, z) \iff$ there is some z' with $\lceil \frac{y-x+1}{2} \rceil \leq z' \leq z$ such that
 359 either $w_{x, x+z'-1} \in L(B)$ and $w_{x+z', y} \in L(C)$, or $w_{x, y-z'} \in L(B)$ and $w_{y-z'+1, y} \in L(C)$.

360 Note that the above equivalences for $\text{Pref}_A^{\text{Maj}}$ and $\text{Suff}_A^{\text{Maj}}$ imply in the particular case
 361 $z = y - x + 1$ the equivalences $\text{Pref}_A^{\text{Maj}}(x, y, z) \iff \text{Suff}_A^{\text{Maj}}(x, y, z) \iff w_{x,y} \in L(A)$.

362 Let us give and justify a list of Horn clauses whose conjunction ψ'_G defines the predicates
 363 $\text{Pref}_A^{\text{Maj}}, \text{Pref}_A^{\text{Min}}, \text{Suff}_A^{\text{Maj}}, \text{Suff}_A^{\text{Min}}$ and Concat_{BC} , using the arithmetic predicates $z = y - x + 1$,
 364 $y - x + 1 = 2z$, and $z \geq \lceil \frac{y-x+1}{2} \rceil$ easily defined in incl-pred-ESO-HORN.

365 **Short rules.** Each rule $A \rightarrow s$ of P is expressed by the two clauses:

- 366 ■ $x = y \wedge z = 1 \wedge Q_s(x) \rightarrow \text{Pref}_A^{\text{Maj}}(x, y, z)$; $x = y \wedge z = 1 \wedge Q_s(x) \rightarrow \text{Suff}_A^{\text{Maj}}(x, y, z)$.

367 **Induction for prefixes.** If we have for $x < y$ the inequalities

368 $\lceil \frac{(y-1)-x+1}{2} \rceil \leq z \leq (y-1) - x + 1$ and $z \geq \lceil \frac{y-x+1}{2} \rceil$ then $\lceil \frac{y-x+1}{2} \rceil \leq z \leq y - x + 1$. This
 369 justifies the clause:

- 370 ■ $x \leq y - 1 \wedge \text{Pref}_A^{\text{Maj}}(x, y - 1, z) \wedge z \geq \lceil \frac{y-x+1}{2} \rceil \rightarrow \text{Pref}_A^{\text{Maj}}(x, y, z)$, for all $A \in N$.

371 For $x < y$ and $y - x + 1 = 2z$, we have $w_{x,x+z-1} = w_{x,y-z}$ and $\lceil \frac{y-x+1}{2} \rceil \leq z \leq y - x$.

372 This justifies the clause:

373 ■ $x \leq y - 1 \wedge \text{Pref}_A^{\text{Maj}}(x, y - 1, z) \wedge y - x + 1 = 2z \rightarrow \text{Pref}_A^{\text{Min}}(x, y, z)$, for all $A \in N$.

374 For $x < y$ and $z > 1$ and $\lceil \frac{(y-1)-x+1}{2} \rceil \leq z - 1 \leq (y - 1) - x$, we have $\lceil \frac{y-x+1}{2} \rceil \leq z \leq y - x$.

375 This justifies the clause:

376 ■ $x \leq y - 1 \wedge z > 1 \wedge \text{Pref}_A^{\text{Min}}(x, y - 1, z - 1) \rightarrow \text{Pref}_A^{\text{Min}}(x, y, z)$, for all $A \in N$.

377 **Induction for suffixes.** As this induction is symmetric to the one for prefixes, we do not
378 justify the following list of induction clauses for the predicates $\text{Suff}_A^{\text{Maj}}$ and $\text{Suff}_A^{\text{Min}}$, $A \in N$:

379 ■ $x + 1 \leq y \wedge \text{Suff}_A^{\text{Maj}}(x + 1, y, z) \wedge z \geq \lceil \frac{y-x+1}{2} \rceil \rightarrow \text{Suff}_A^{\text{Maj}}(x, y, z)$;

380 ■ $x + 1 \leq y \wedge \text{Suff}_A^{\text{Maj}}(x + 1, y, z) \wedge y - x + 1 = 2z \rightarrow \text{Suff}_A^{\text{Min}}(x, y, z)$;

381 ■ $x + 1 \leq y \wedge z > 1 \wedge \text{Suff}_A^{\text{Min}}(x + 1, y, z - 1) \rightarrow \text{Suff}_A^{\text{Min}}(x, y, z)$.

382 **Concatenation.** For all $B, C \in N$, it is clear that the concatenation predicate Concat_{BC}
383 is defined inductively by the following three clauses:

384 ■ *initialization:* $\text{Pref}_B^{\text{Maj}}(x, y, z) \wedge \text{Suff}_C^{\text{Min}}(x, y, z) \rightarrow \text{Concat}_{BC}(x, y, z)$;

385 $\text{Pref}_B^{\text{Min}}(x, y, z) \wedge \text{Suff}_C^{\text{Maj}}(x, y, z) \rightarrow \text{Concat}_{BC}(x, y, z)$;

386 ■ *induction:* $z > 1 \wedge \text{Concat}_{BC}(x, y, z - 1) \rightarrow \text{Concat}_{BC}(x, y, z)$.

387 **Long rules.** Each rule $A \rightarrow B_1C_1 \& \dots \& B_mC_m$ of P is expressed by the two clauses:

388 ■ $z = y - x + 1 \wedge \text{Concat}_{B_1C_1}(x, y, z) \wedge \dots \wedge \text{Concat}_{B_mC_m}(x, y, z) \rightarrow \text{Pref}_A^{\text{Maj}}(x, y, z)$;

389 ■ $z = y - x + 1 \wedge \text{Concat}_{B_1C_1}(x, y, z) \wedge \dots \wedge \text{Concat}_{B_mC_m}(x, y, z) \rightarrow \text{Suff}_A^{\text{Maj}}(x, y, z)$.

390 Thus, the formula $\forall x \forall y \forall z \psi'_G$ where ψ'_G is the conjunction of the above clauses defines the
391 predicates $\text{Pref}_A^{\text{Maj}}$, $\text{Pref}_A^{\text{Min}}$, $\text{Suff}_A^{\text{Maj}}$, $\text{Suff}_A^{\text{Min}}$, and Concat_{BC} .

392 **Definition of $\Sigma^+ \setminus L$.** We have the equivalence $\text{Pref}_S^{\text{Maj}}(1, n, n) \iff w \in L(S) \iff w \in L$.

393 Therefore, the following contradiction clause expresses $w \notin L$:

394 ■ $\gamma_S := \min(x) \wedge \max(y) \wedge \max(z) \wedge \text{Pref}_S^{\text{Maj}}(x, y, z) \rightarrow \perp$.

395 Finally, observe that the formula $\Phi_G := \exists R \forall x \forall y \forall z \psi_G$ where ψ_G is $\gamma_{\text{arith}} \wedge \psi'_G \wedge \gamma_S$
396 and γ_{arith} is the conjunction of clauses that define the arithmetic predicates, belongs to
397 **incl-pred-ESO-HORN**. Since we have $\langle w \rangle \models \Phi_G \iff w \notin L$, as justified above, then the
398 langage $\Sigma^+ \setminus L$ belongs to **incl-pred-ESO-HORN**, as claimed. ◀

399 4.2 Equivalence of logic with cube-circuits

400 We now introduce the *cube-circuit*, an extension of the grid-circuit to three dimensions. It
401 will make the link between our logic **incl-pred-ESO-HORN** and the class **RealTime2OCA**.

402 ▶ **Definition 19.** A cube-circuit is a tuple $C := (\Sigma, (\text{Input}_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{\text{acc}}, \mathbf{g})$ where

403 ■ Σ is the input alphabet and $(\text{Input}_n)_{n>0}$ is the family of input functions

404 $\text{Input}_n : \Sigma^n \times [1, n]^3 \rightarrow \Sigma \cup \{\$\}$ such that, for $w = w_1 \dots w_n \in \Sigma^n$,

405 $\text{Input}_n(w, x, y, z) = w_y$ if $x = y$ and $z = 1$, and $\text{Input}_n(w, x, y, z) = \$$ otherwise,

406 ■ $\mathbf{Q} \cup \{\#\}$ is the finite set of states and $\mathbf{Q}_{\text{acc}} \subseteq \mathbf{Q}$ is the subset of accepting states,

407 ■ $\mathbf{g} : (\mathbf{Q} \cup \{\#\})^3 \times (\Sigma \cup \{\$\}) \rightarrow \mathbf{Q}$ is the transition function.

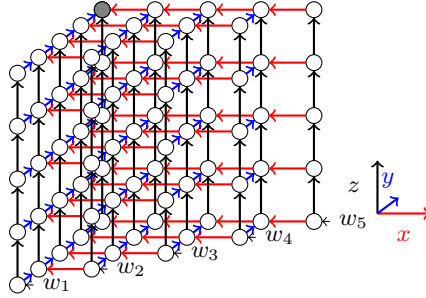
408 ▶ **Definition 20** (computation of a cube-circuit). The computation C_w of a cube-circuit
409 $C := (\Sigma, (\text{Input}_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{\text{acc}}, \mathbf{g})$ on a word $w = w_1 \dots w_n \in \Sigma^n$ is a grid of $(n + 1)^3$ sites
410 $(x, y, z) \in [1, n + 1] \times [0, n]^2$, each in a state $\langle x, y, z \rangle \in \mathbf{Q} \cup \{\#\}$ computed inductively:

411 ■ each site (x, y, z) such that $x > y$ or $z = 0$ is in the state $\#$;

XX:12 Conjunctive grammars, cellular automata and logic

412 ■ the state of each site $(x, y, z) \in [1, n]^3$ such that $x \leq y$ and $z > 0$ is
 413 $\langle x, y \rangle = \mathbf{g}(\langle x + 1, y, z \rangle, \langle x, y - 1, z \rangle, \langle x, y, z - 1 \rangle, \text{Input}_n(w, x, y, z))$.

414 A word $w = w_1 \dots w_n \in \Sigma^n$ is *accepted* by the cube-circuit C if the output state $\langle 1, n, n \rangle$
 415 of C_w belongs to \mathbf{Q}_{acc} . The language *recognized* by C is the set of words it accepts. We
 416 denote by **Cube** the class of languages recognized by a cube-circuit.



■ **Figure 6** The cube-circuit

417 Actually, the logic **incl-pred-ESO-HORN** is equivalent to cube-circuits.

418 ► **Lemma 21.** **incl-pred-ESO-HORN** = **Cube**.

419 **Proof.** The proof is similar to that of Lemma 14. The cube-circuit can be seen as the
 420 “normalized form” of a formula of **incl-pred-ESO-HORN**, proving the inclusion **Cube** \subseteq
 421 **incl-pred-ESO-HORN**. The proof of inverse inclusion is divided into the same three steps
 422 as for Lemma 14, which are easily adaptable to three variables: elimination of atoms
 423 $R(x + a, y - b, z - c)$ for $a + b + c > 1$, elimination of hypotheses $R(x, y, z)$ and transformation
 424 of the resulting formula into a cube-circuit. ◀

4.3 Cube-circuits are equivalent to real-time 2-OCA

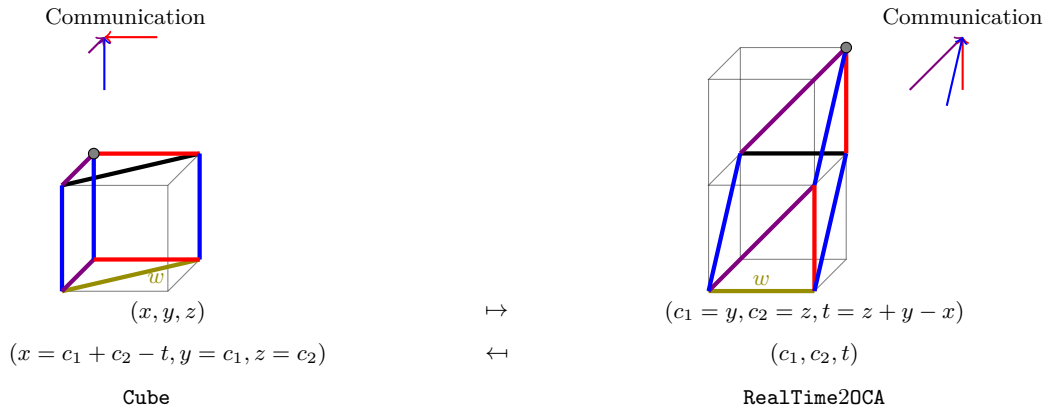
426 One observes that by a one-to-one transformation, the computation C_w of a cube-circuit C
 427 on a word w is nothing else than the space-time diagram of a real-time 2-OCA on the input w .
 428 This yields:

429 ► **Lemma 22.** **Cube** = **RealTime2OCA**.

430 **Proof.** The bijection between the sites (x, y, z) of the computation C_w of a cube-circuit C
 431 on a word w and the sites (c_1, c_2, t) of the space-time diagram of a real-time 2-OCA on the
 432 input w is depicted in Figure 7. We check that this bijection respects the communication
 433 scheme and the input/output sites of both computation models as shown in Figure 7.
 434 By this transformation, the transition function \mathbf{g} of the cube-circuit, which is $\langle x, y, z \rangle =$
 435 $\mathbf{g}(\langle x + 1, y, z \rangle, \langle x, y - 1, z \rangle, \langle x, y, z - 1 \rangle, \text{Input}_n(w, x, y, z))$ becomes the transition function
 436 \mathbf{f} of the 2-OCA: $\langle c_1, c_2, t \rangle = \mathbf{f}(\langle c_1, c_2, t - 1 \rangle, \langle c_1 - 1, c_2, t - 1 \rangle, \langle c_1, c_2 - 1, t - 1 \rangle)$, and vice
 437 versa. ◀

438
 439 **Proof of Theorem 16.** Lemmas 21 and 22 give us the following equalities of classes:
 440 **incl-pred-ESO-HORN** = **Cube** = **RealTime2OCA**.

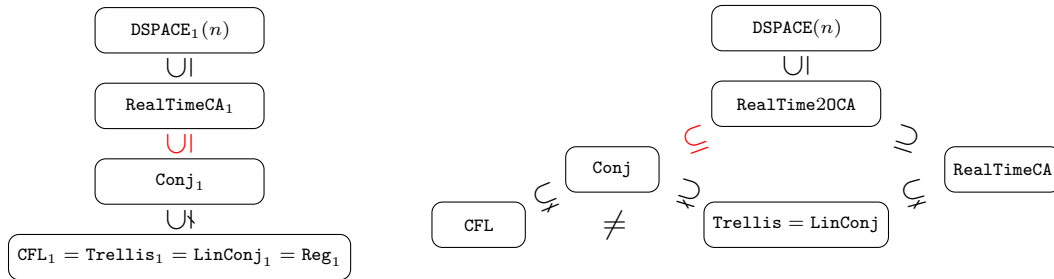
441 From the fact that the class **RealTime2OCA** is closed under complement and from
 442 Lemma 18, we deduce **Conj** \subseteq **incl-pred-ESO-HORN** = **Cube** = **RealTime2OCA**. ◀



■ **Figure 7** Bijection between the sites of C_w and the space-time sites of a 2-OCA on w

443 **5 Conclusion**

444 We have proved the inclusions $\text{Conj}_1 \subseteq \text{RealTimeCA}$ and $\text{Conj} \subseteq \text{RealTime2OCA}$ by expressing
 445 in two logics (proved equivalent to RealTimeCA and RealTime2OCA , respectively) the inductive
 446 process of a conjunctive grammar. Figure 8 recapitulates the known inclusions between the
 447 language classes that we have considered here. To grasp the expressive power of the Conj
 448 (resp. Conj_1) class, it would be important to obtain exact characterizations of this class in
 449 logic and/or computational complexity.



■ **Figure 8** Relations between language classes over a unary or general alphabet

450 **Acknowledgments:** This paper would not exist without the inspiration of Véronique Terrier.
 451 Her in-depth knowledge of cellular automata and their complexity classes, the references and
 452 advice she generously gave us, as well as her careful reading, were essential in designing and
 453 finalizing the results and the presentation of the paper. E.g., the class diagram of Figure 8 is
 454 due to her. This work has been partly supported by the PING/ACK project of the French
 455 National Agency for Research (ANR-18-CE40-0011).

456 **References**

457 1 Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley,
 458 1995.
 459 2 Nicolas Bacquey, Etienne Grandjean, and Frédéric Olive. Definability by Horn Formulas and
 460 Linear Time on Cellular Automata. In *ICALP 2017*, volume 80, pages 99:1–99:14, 2017.
 461 3 Egon Börger, Erich Grädel, and Yuri Gurevich. *The Classical Decision Problem*. Perspectives
 462 in Mathematical Logic. Springer, 1997.

XX:14 Conjunctive grammars, cellular automata and logic

- 463 4 Jik H. Chang, Oscar H. Ibarra, and Michael A. Palis. Efficient simulations of simple models of
464 parallel computation by time-bounded atms and space-bounded tms. *Theor. Comput. Sci.*,
465 68(1):19–36, 1989.
- 466 5 Marianne Delorme and Jacques Mazoyer. *Cellular Automata as Language Recognizers in
467 Cellular Automata: a Parallel Model*. Kluwer, 1999.
- 468 6 Ronald Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In
469 *Complexity of Computation, SIAM-AMS Proceedings*, pages 43–73, 1974.
- 470 7 Dora Giammarresi, Antonio Restivo, Sebastian Seibert, and Wolfgang Thomas. Monadic
471 second-order logic over rectangular pictures and recognizability by tiling systems. *Inf. Comput.*,
472 125(1):32–45, 1996.
- 473 8 Erich Grädel. Capturing complexity classes by fragments of second-order logic. *Theoretical
474 Computer Science*, 101(1):35–57, 1992.
- 475 9 Erich Grädel, Phokion G. Kolaitis, Leonid Libkin, Maarten Marx, Joel Spencer, Moshe Y.
476 Vardi, Yde Venema, and Scott Weinstein. *Finite Model Theory and Its Applications*. Springer,
477 2007.
- 478 10 Etienne Grandjean and Théo Grente. Descriptive complexity for minimal time of cellular
479 automata. In *LICS, 2019*, pages 1–13, 2019.
- 480 11 Etienne Grandjean, Théo Grente, and Véronique Terrier. Inductive definitions in logic versus
481 programs of real-time cellular automata. hal.archives-ouvertes.fr/hal-02474520/ submitted to
482 *Theoretical Computer Science*, 62 pages, February 2020.
- 483 12 Etienne Grandjean and Frédéric Olive. A logical approach to locality in pictures languages.
484 *Journal of Computer and System Science*, 82(6):959–1006, 2016.
- 485 13 Oscar H. Ibarra and Tao Jiang. Relating the power of cellular arrays to their closure properties.
486 *Theor. Comput. Sci.*, 57:225–238, 1988.
- 487 14 Neil Immerman. *Descriptive complexity*. Springer, 1999.
- 488 15 Artur Jez. Conjunctive grammars generate non-regular unary languages. *Int. J. Found.
489 Comput. Sci.*, 19(3):597–615, 2008.
- 490 16 K. N. King. Alternating multihead finite automata. *Theor. Comput. Sci.*, 61:149–174, 1988.
- 491 17 Hans Kleine Büning and Theodor Lettmann. *Propositional logic - deduction and algorithms*,
492 volume 48 of *Cambridge tracts in theoretical computer science*. Cambridge University Press,
493 1999.
- 494 18 S. Rao Kosaraju. Speed of recognition of context-free languages by array automata. *SIAM J.
495 Comput.*, 4(3):331–340, 1975.
- 496 19 Leonid Libkin. *Elements of Finite Model Theory*. Texts in Theoretical Computer Science. An
497 EATCS Series. Springer, 2004.
- 498 20 Alexander Okhotin. Conjunctive grammars. *J. Autom. Lang. Comb.*, 6(4):519–535, 2001.
- 499 21 Alexander Okhotin. Boolean grammars. *Information and Computation*, 194(1):19 – 48, 2004.
- 500 22 Alexander Okhotin. On the equivalence of linear conjunctive grammars and trellis automata.
501 *Theoretical Informatics and Applications*, 38(1):69–88, 2004.
- 502 23 Alexander Okhotin. Conjunctive and boolean grammars: The true general case of the
503 context-free grammars. *Computer Science Review*, 9:27–59, 2013.
- 504 24 Véronique Terrier. Closure properties of cellular automata. *Theor. Comput. Sci.*, 352(1-3):97–
505 107, 2006.
- 506 25 Véronique Terrier. Low complexity classes of multidimensional cellular automata. *Theor.
507 Comput. Sci.*, 369(1-3):142–156, 2006.
- 508 26 Véronique Terrier. Language recognition by cellular automata. In *Handbook of Natural
509 Computing*, pages 123–158. Springer, 2012.
- 510 27 Hao Wang. Dominoes and the aea case of the decision problem. In *Proceedings on the
511 Symposium on the Mathematical Theory of Automata, April 1962*, pages 23–55, 1963.

512 **Appendix A: Complement of proof for Lemma 14**

513 **Elimination of hypotheses** $R(x, y)$: The first idea is to group together in each computation
 514 clause the hypothesis atoms of the form $R(x, y)$ and the conclusion of the clause. Accordingly,
 515 the formula obtained Φ can be rewritten in the form

$$516 \quad \Phi := \exists \mathbf{R} \forall x \forall y \left[\bigwedge_i C_i(x, y) \wedge \bigwedge_{i \in [1, k]} (\alpha_i(x, y) \rightarrow \theta_i(x, y)) \right]$$

517 where the C_i 's are the input clauses and the contradiction clause and each computation
 518 clause is written in the form $\alpha_i(x, y) \rightarrow \theta_i(x, y)$ where $\alpha_i(x, y)$ is a conjunction of formulas
 519 of the only forms $R(x-1, y) \wedge \neg \mathbf{min}(x)$, $R(x, y-1) \wedge \neg \mathbf{min}(y)$ (but not $R(x, y)$), and $\theta_i(x, y)$
 520 is a Horn clause whose *all* atoms are of the form $R(x, y)$.

521 We number R_1, \dots, R_m the computation predicates of \mathbf{R} . To each subset $J \subseteq [1, k]$ of
 522 the family of implications $(\alpha_i(x, y) \rightarrow \theta_i(x, y))_{i \in [1, k]}$ let us associate the set

$$523 \quad K_J := \{h \in [1, m] \mid \bigwedge_{i \in J} \theta_i(x, y) \rightarrow R_h(x, y) \text{ is a tautology}\}.$$

524 Note that the notion of *tautology* used in the definition of K_J is “propositional” because all
 525 the atoms involved are of the form $R_i(x, y)$, i.e., refer to the same pair of variables (x, y) .
 526 Also, note that the function $J \mapsto K_J$ is *monotonic*: for $J' \subseteq J$, we have $K_{J'} \subseteq K_J$ because
 527 $\bigwedge_{i \in J'} \theta_i(x, y) \rightarrow R_h(x, y)$ implies $\bigwedge_{i \in J} \theta_i(x, y) \rightarrow R_h(x, y)$.

528 Clearly, it is enough to prove the following claim:

529 \triangleright **Claim 23.** The formula Φ is equivalent to the following formula Φ' , whose clauses have *no*
 530 *hypothesis* $R(x, y)$.

$$531 \quad \Phi' := \exists \mathbf{R} \forall x \forall y \left[\bigwedge_i C_i(x, y) \wedge \bigwedge_{J \subseteq [1, k]} \bigwedge_{h \in K_J} \left(\bigwedge_{i \in J} \alpha_i(x, y) \rightarrow R_h(x, y) \right) \right]$$

532 *Proof of the implication* $\Phi \Rightarrow \Phi'$: It is enough to prove the implication

$$533 \quad \left[\bigwedge_{i \in [1, k]} (\alpha_i(x, y) \rightarrow \theta_i(x, y)) \right] \rightarrow \left[\bigwedge_{i \in J} \alpha_i(x, y) \rightarrow \bigwedge_{h \in K_J} R_h(x, y) \right]$$

534 for all set $J \subseteq [1, k]$. The implication to be proved can be equivalently written:

$$535 \quad \left[\bigwedge_{i \in J} \alpha_i(x, y) \wedge \bigwedge_{i \in [1, k]} (\alpha_i(x, y) \rightarrow \theta_i(x, y)) \right] \rightarrow \bigwedge_{h \in K_J} R_h(x, y).$$

536 The sub-formula between brackets above implies the conjunction $\bigwedge_{i \in J} \theta_i(x, y)$. As the implic-
 537 ation $\bigwedge_{i \in J} \theta_i(x, y) \rightarrow \bigwedge_{h \in K_J} R_h(x, y)$ is a tautology (by definition of K_J), the implication
 538 to be proved is a tautology too.

539 The converse implication $\Phi' \Rightarrow \Phi$ is more difficult to prove. It uses a folklore property of
 540 propositional Horn formulas easy to be proved:

541 \blacktriangleright **Lemma 24** (Horn property: folklore). *Let F be a strict Horn formula of propositional*
 542 *calculus, that is a conjunction of clauses of the form $p_1 \wedge \dots \wedge p_k \rightarrow p_0$ where $k \geq 0$ and the*

XX:16 Conjunctive grammars, cellular automata and logic

543 p_i 's are propositional variables. Let F' be the conjunction of propositional variables q such
 544 that the implication $F \rightarrow q$ is a tautology. F has the same minimal model⁵ as F' .

545 *Proof of the implication $\Phi' \Rightarrow \Phi$:* Let $\langle w \rangle$ be a model of Φ' and let $(\langle w \rangle, \mathbf{R})$ be the minimal
 546 model of the Horn formula

$$547 \quad \varphi' := \forall x \forall y \left[\bigwedge_i C_i(x, y) \wedge \bigwedge_{J \subseteq [1, k]} \bigwedge_{h \in K_J} \left(\bigwedge_{i \in J} \alpha_i(x, y) \rightarrow R_h(x, y) \right) \right].$$

548 It is enough to show that $(\langle w \rangle, \mathbf{R})$ also satisfies the formula

$$549 \quad \varphi := \forall x \forall y \left[\bigwedge_i C_i(x, y) \wedge \bigwedge_{i \in [1, k]} (\alpha_i(x, y) \rightarrow \theta_i(x, y)) \right].$$

550 As each α_i is a conjunction of formulas of the form $R(x-1, y) \wedge \neg \min(x)$, or
 551 $R(x, y-1) \wedge \neg \min(y)$, we make an induction on the domain $\{(a, b) \in [1, n]^2 \mid a + b \leq t\}$, for
 552 $t \in [1, 2n]$. More precisely, we are going to prove, by recurrence on the integer $t \in [1, 2n]$,
 553 that the minimal model $(\langle w \rangle, \mathbf{R})$ of φ' satisfies the “relativized” formula φ_t of the formula φ
 554 defined by

$$555 \quad \varphi_t := \forall x \forall y \left[x + y \leq t \rightarrow \left[\bigwedge_i C_i(x, y) \wedge \bigwedge_{i \in [1, k]} (\alpha_i(x, y) \rightarrow \theta_i(x, y)) \right] \right]$$

556 As the hypothesis $x + y \leq 2n$ holds for all x, y in the domain $[1, n]$, φ_{2n} is equivalent to φ on
 557 the structure $(\langle w \rangle, \mathbf{R})$.

558 *Basis case:* For $t = 1$ the set $\{(a, b) \in [1, n]^2 \mid a + b \leq t\}$ is empty so that the “relativized”
 559 formula φ_1 is trivially true in the minimal model $(\langle w \rangle, \mathbf{R})$ of φ' .

560 *Recurrence step:* Suppose $(\langle w \rangle, \mathbf{R}) \models \varphi_{t-1}$, for an integer $t \in [2, 2n]$. It is enough to show
 561 that, for each couple $(a, b) \in [1, n]^2$ such that $a + b = t$, we have $(\langle w \rangle, \mathbf{R}) \models \bigwedge_{i \in [1, k]} (\alpha_i(a, b) \rightarrow$
 562 $\theta_i(a, b))$. Let $J_{a,b}$ be the set of indices $i \in [1, k]$ such that the couple (a, b) satisfies α_i :

$$563 \quad J_{a,b} := \{i \in [1, k] \mid (\langle w \rangle, \mathbf{R}) \models \alpha_i(a, b)\}.$$

564 Recall that each $\alpha_i(a, b)$ is a (possibly empty) conjunction of atoms $R(a', b')$ with $(a', b') =$
 565 $(a-1, b)$ or $(a', b') = (a, b-1)$, therefore such that $a' + b' = t-1$. Let $J \subseteq [1, k]$ be any set.
 566 Let us examine the two possible cases:

567 1) $J \subseteq J_{a,b}$: then the conjunction $\bigwedge_{i \in J} \alpha_i(a, b)$ holds in $(\langle w \rangle, \mathbf{R})$; hence, in $(\langle w \rangle, \mathbf{R})$, the
 568 conjunction $\bigwedge_{h \in K_J} (\bigwedge_{i \in J} \alpha_i(a, b) \rightarrow R_h(a, b))$ is equivalent to $\bigwedge_{h \in K_J} R_h(a, b)$;

569 2) $J \setminus J_{a,b} \neq \emptyset$: then the conjunction $\bigwedge_{i \in J} \alpha_i(a, b)$ is false in $(\langle w \rangle, \mathbf{R})$; hence, the
 570 conjunction $\bigwedge_{h \in K_J} (\bigwedge_{i \in J} \alpha_i(a, b) \rightarrow R_h(a, b))$ holds in $(\langle w \rangle, \mathbf{R})$.

571 From (1) and (2), we deduce that in $(\langle w \rangle, \mathbf{R})$ the conjunction $\bigwedge_{J \subseteq [1, k]} \bigwedge_{h \in K_J} (\bigwedge_{i \in J} \alpha_i(a, b) \rightarrow$
 572 $R_h(a, b))$ is equivalent to the conjunction $\bigwedge_{J \subseteq J_{a,b}} \bigwedge_{h \in K_J} R_h(a, b)$, which can be simplified
 573 as $\bigwedge_{h \in K_{J_{a,b}}} R_h(a, b)$ because $J \subseteq J_{a,b}$ implies $K_J \subseteq K_{J_{a,b}}$. Consequently, for all $h \in [1, m]$,
 574 the minimal model $(\langle w \rangle, \mathbf{R})$ of the Horn formula φ' satisfies the atom $R_h(a, b)$ iff h belongs
 575 to $K_{J_{a,b}}$. By definition,

⁵ For example, for $F := p_1 \wedge p_3 \wedge (p_1 \wedge p_3 \rightarrow p_5) \wedge (p_1 \wedge p_2 \rightarrow p_4)$, we have $F' := p_1 \wedge p_3 \wedge p_5$, which has the same minimal model I as F ; this model is given by $I(p_1) = I(p_3) = I(p_5) = 1$ and $I(p_2) = I(p_4) = 0$.

576 $K_{J_{a,b}} := \{h \in [1, m] \mid \bigwedge_{i \in J_{a,b}} \theta_i(x, y) \rightarrow R_h(x, y) \text{ is a tautology}\}$

577 or, equivalently,

578 $K_{J_{a,b}} := \{h \in [1, m] \mid \bigwedge_{i \in J_{a,b}} \theta_i(a, b) \rightarrow R_h(a, b) \text{ is a tautology}\}.$

579 As a consequence of Lemma 24, the two conjunctions

$$580 \quad \bigwedge_{i \in J_{a,b}} \theta_i(a, b) \text{ and } \bigwedge_{h \in K_{J_{a,b}}} R_h(a, b)$$

581 have the same minimal model, which is also the restriction of the minimal model $(\langle w \rangle, \mathbf{R})$ of
 582 φ' to the set of atoms $R_h(a, b)$, for $h \in [1, m]$. Therefore, if $i \in J_{a,b}$, then $(\langle w \rangle, \mathbf{R}) \models \theta_i(a, b)$.
 583 If $i \in [1, k] \setminus J_{a,b}$, then we have $(\langle w \rangle, \mathbf{R}) \models \neg \alpha_i(a, b)$, by definition of $J_{a,b}$. Therefore, for
 584 all $i \in [1, k]$, we get $(\langle w \rangle, \mathbf{R}) \models \neg \alpha_i(a, b) \vee \theta_i(a, b)$. In other words, for all (a, b) such that
 585 $a + b = t$, we have : $(\langle w \rangle, \mathbf{R}) \models \bigwedge_{i \in [1, k]} (\alpha_i(a, b) \rightarrow \theta_i(a, b))$ and then $(\langle w \rangle, \mathbf{R}) \models \varphi_t$.

586 This concludes the inductive proof that $(\langle w \rangle, \mathbf{R}) \models \varphi_t$, for all $t \in [1, 2n]$, and then
 587 $\langle w \rangle \models \Phi$. This proves the converse implication $\Phi' \Rightarrow \Phi$. Claim 23 is demonstrated. \square

588 Appendix B: Complement of proof for Lemma 15

589 **Grid \subseteq RealTimeCA.** To prove this inclusion, we show how to simulate the computation of
 590 the grid-circuit on a real-time CA. The simulation is made by a geometric transformation that
 591 embeds the grid-circuit in the space-time diagram of a real-time CA. This transformation is
 592 divided into three steps:

- 593 1. a variable change: we apply to each site $(x, y) \in [1, n]^2$ of the grid-circuit the variable
 594 change $(x, y) \mapsto (c' = y - x + 1, t' = x + y - 1)$;
- 595 2. a folding: we fold the resulting diagram along the axis $c' = 1$: each site (c', t') with $c' < 1$
 596 is sent to its symmetric counterpart $(-c' + 1, t')$;
- 597 3. a grouping: each site $(c, t) = (\lceil \frac{c'}{2} \rceil, \lceil \frac{t'}{2} \rceil)$ of the new diagram records the set of sites
 598 $\{(c' - 1, t' - 1), (c', t'), (c' + 1, t' - 1)\}$ with c' and t' odd and greater than 1.

599 The resulting diagram is the expected space-time diagram of a real-time CA, proving the
 600 inclusion.

601 **RealTimeCA \subseteq Grid.** To simulate a real-time CA $\mathcal{A} = (\mathbf{S}, \mathbf{S}_{\text{accept}}, \{-1, 0, 1\}, \mathbf{f})$ on the grid,
 602 we first turn \mathcal{A} into an equivalent CA $\mathcal{A}' = (\mathbf{S}, \mathbf{S}_{\text{accept}}, \{-2, -1, 0\}, \mathbf{f})$. This transformation
 603 can be seen as the variable change $(c, t) \mapsto (c + t - 1, t)$. The diagram of \mathcal{A}' is then embedded
 604 on the grid-circuit C' by applying to its sites (c', t') the variable change $(c', t') \mapsto (t', c')$. The
 605 local and uniform communication of the embedded diagram can easily be carried out by the
 606 grid-circuit communication scheme.