

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. halo3167529v1

HAL Id: hal-03167529 https://hal.science/hal-03167529v1

Preprint submitted on 12 Mar 2021 (v1), last revised 9 Nov 2022 (v2)

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

₃ Théo Grente ⊠

4 GREYC, Université de Caen Normandie, France

🛿 Étienne Grandjean 🖂

6 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 Conj \subseteq ? RealTimeCA: 1) it is true for unary languages; 2) Conj \subseteq RealTime20CA, 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 RealTime20CA.

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

17 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 33 subclass of languages generated by *linear* conjunctive grammars was found to be equal to 34 the Trellis class of languages recognized by *trellis* automata [22], or equivalently, *one-way* 35 *real-time* cellular automata. Faced with this result, it is tempting to ask the following 36 question: is the larger class Conj equal to the class RealTimeCA of languages recognized by 37 two-way real-time cellular automata? Either answer to this question has strong consequences: 38 **If Conj** = **RealTimeCA** then each of the two classes will benefit from the closure properties 39 of the other class; in particular, RealTimeCA would be closed under reverse, which 40 was shown by [13] to imply RealTimeCA = LinearTime_{CA}, i.e. real-time is nothing but 41 linear time for cellular automata, a surprising positive answer to a longstanding open 42 question [5, 24, 26]. 43 If $\texttt{Conj} \neq \texttt{RealTimeCA}$ then $\texttt{Conj} \subsetneq \texttt{DSPACE}(n)$ or $\texttt{RealTimeCA} \subsetneq \texttt{DSPACE}(n)$: any of these 44

45 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

XX:2 Conjunctive grammars, cellular automata and logic

Real-time is the minimal time of cellular automata (CA). Recall that RealTimeCA (resp. 46 Trellis) is the class of languages recognized in real-time by one-dimensional CA with two-way 47 (resp. one-way) communication and input word given in parallel. We know the strict inclusion 48 Trellis \subseteq RealTimeCA. The robustness of these classes is attested by their characteriz-49 ation by two sub-logics of ESO – the existential second-order logic, which characterizes 50 NP - with Horn formulas as their first-order parts¹, and called respectively pred-ESO-HORN 51 and incl-ESO-HORN, see [10, 11]. For short, we write RealTimeCA = pred-ESO-HORN and 52 Trellis = incl-ESO-HORN. 53 **Results of this paper:** This paper focuses on the relationships between the class of 54

⁵⁴ **Itestitis 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 Conj =? RealTimeCA or even to the question of the inclusion ⁵⁷ Conj \subseteq ? RealTimeCA, we prove two weakened versions of this inclusion:

⁵⁸ 1. $Conj_1 \subseteq RealTimeCA_1$: The inclusion holds when restricted to *unary* languages².

⁵⁹ 2. Conj \subseteq RealTime20CA: The inclusion holds for real-time of *two-dimensional one-way* ⁶⁰ cellular automata (2-OCA). (We have RealTimeCA \subseteq RealTime20CA.)

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

⁶⁵ Our inclusion (2) improves the inclusion $CFL \subseteq RealTime2SOCA$, 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 CFL \subseteq Conj \subseteq RealTime2OCA \subseteq RealTime2SOCA.

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

In this paper, the same logic programming method is adopted. We prove inclusion (1), Conj₁ \subseteq RealTimeCA₁, by expressing a unary language generated by a conjunctive grammar in the pred-ESO-HORN logic. Inclusion (1) follows, by the equality pred-ESO-HORN = RealTimeCA. Similarly, to prove inclusion (2), Conj \subseteq RealTime2OCA, we first design a logic denoted incl-pred-ESO-HORN so that incl-pred-ESO-HORN = RealTime2OCA. Then, we express any conjunctive language in this logic, proving that it belongs to RealTime2OCA, as claimed.

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

Our proof method and the paper structure: After Section 2 gives some definitions,
Sections 3 and 4 present inclusions (1) and (2) and their proofs with a common plan:
Subsection 3.1 (resp. 4.1) expresses the inductive generating process of a conjunctive
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 C is denoted C_1 .

⁹⁰ incl-pred-ESO-HORN). Subsection 3.2 (resp. 4.2) shows that any formula of this logic can

⁹¹ be normalized into a formula which mimics the computation of a two-dimensional (resp.

⁹² three-dimensional) grid-circuit called Grid (resp. Cube); Subsection 3.3 (resp. 4.3) trans-

⁹³ lates 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:
 pred-ESO-HORN = Grid = RealTimeCA and incl-pred-ESO-HORN = Cube = RealTime2OCA.

⁹⁵ pred-ESO-HORN = Grid = RealTimeCA and incl-pred-ESO-HORN = Cube = RealTime20CA.
 ⁹⁶ Section 5 gives a conclusion with a diagram of the known relations between the Conj class

⁹⁷ and the CA complexity classes studied here, for the general case and for the unary case.

⁹⁸ 2 Preliminaries

⁹⁹ 2.1 Conjunctive grammars and their binary normal form

¹⁰⁰ 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 $A \to \alpha_1 \& ... \& \alpha_m$, for $m \ge 1$ 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 $A \to \alpha_1^1 \& ... \& \alpha_{m_1}^1 | \cdots | \alpha_1^k \& ... \& \alpha_{m_k}^k$, 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: $A \to B_1C_1$ &...& B_mC_m ($m \ge 1, B_i, C_j \in N$);

113 \blacksquare a short rule: $A \to a \ (a \in \Sigma)$.

114 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

 $(w) := ([1, n]; (Q_s)_{s \in \Sigma}, \min, \max, \text{suc}, \text{pred})$

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 S_{Σ} denote the signature $\{(Q_s)_{s\in\Sigma}, \min, \max, \text{suc}, \text{pred}\}\$ of the structure $\langle w \rangle$.

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

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

128 2.3 Cellular automata and real-time

▶ Definition 4 (1-CA and 2-0CA). A *d*-dimensional cellular automaton (CA) is a triple (S, \mathcal{N} , f) where S is the finite set of states, $\mathcal{N} \subset \mathbb{Z}^d$ is the neighborhood, and $\mathbf{f} : \mathbf{S}^{|\mathcal{N}|} \to \mathbf{S}$ is

¹³¹ the transition function. We are interested in the following two special cases:

- ¹³² = 1-CA: It is a one-dimensional two-way cellular automaton $(S, \{-1, 0, 1\}, f)$, 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 $(S, \{(0,0), (-1,0), (0,-1)\}, f)$ 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 $\langle c_1, c_2, t \rangle = \mathbf{f}(\langle c_1, c_2, t-1 \rangle, \langle c_1 1, c_2, t-1 \rangle, \langle c_1, c_2 1, t-1 \rangle).$

Definition 5 (permanent and quiescent states). In a CA, a state \sharp is permanent if a cell in state \sharp remains in this state forever. A state λ of a CA is quiescent if a cell in state λ 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 (S, N, f) with an input ¹⁴² alphabet $\Sigma \subset S$, a permanent state \sharp , a quiescent state λ , and a set of accepting states ¹⁴³ S_{acc} \subset 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).

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 initial time 1: $\langle i, 1 \rangle = w_i$. All other cells are in the permanent state \sharp . For a 2-OCA, the *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 $\langle c_1, c_2 \rangle \in [1, n] \times [2, n]$ 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).
- Acceptance. An input word is accepted by a 1-CA (resp. 2-CA) at time t if the output cell
 enters an accepting state at time t.



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

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

3 Real-time recognition of a unary conjunctive language

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

Theorem 8. $Conj_1 \subseteq RealTimeCA_1$.



Figure 2 Space-time diagrams of RealTimeCA and RealTime20CA

¹⁶¹ 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 $\Phi := \exists \mathbf{R} \forall x \forall y \psi(x, y)$ 164 where **R** is a finite set of binary predicates and ψ is a conjunction of Horn clauses, of signature 165 $S_{\Sigma} \cup \mathbf{R}$, and of one the three following forms: 166 an input clause: $\min(x) \land (\neg) \min(y) \land Q_s(y) \to R(x, y)$ with $s \in \Sigma$ and $R \in \mathbf{R}$; 167 a computation clause: $\delta_1 \wedge \ldots \wedge \delta_r \to R(x, y)$ with $R \in \mathbf{R}$ and where each hypothesis δ_i is 168 an atom S(x, y) or a conjunction $S(x - a, y - b) \land x > a \land y > b$, with $S \in \mathbf{R}$ and $a, b \ge 0$ 169 two integers such that a + b > 0: 170 a contradiction clause: $\max(x) \wedge \max(y) \wedge R(x, y) \rightarrow \bot$ with $R \in \mathbf{R}$. 171

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: $\min(x) \to R^{x=1}(x, y); x > 1 \land R^{x=1}(x-1, y) \to R^{x=2}(x, y);$

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

¹⁷⁹ Also, some other arithmetic predicates easily defined in pred-ESO-HORN will be used. For ¹⁸⁰ example, y = 2x can be replaced by the atom $R^{y=2x}(x, y)$, where $R^{y=2x}$ is defined by the ¹⁸¹ following two clauses using the predicates $R^{x=1}$, $R^{y=2}$, $R^{x>1}$ and $R^{y>2}$:

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

▶ 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}_{arith} , which consists of $R^{x=y}$, $R^{y=2x}$ and $R^{\rho(x,y)}$, for $\rho(x,y) \coloneqq x \ge \lfloor \frac{y}{2} \rfloor$, 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-HORN₁.

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

¹⁹⁰ **Proof.** Let $G = (\{a\}, N, P, S)$ be a conjunctive grammar in binary normal form which ¹⁹¹ generates L. For each $A \in N$ and each unary word a^y , we have, according to the length y, ¹⁹² the following equivalences which will be the basis of our induction: if y = 1, then $a^y \in L(A) \iff$ the short rule $A \to a$ belongs to P;

¹⁹⁴ if y > 1, then $a^y \in L(A) \iff$ there is a long rule $A \to B_1C_1\&\ldots\&B_mC_m$ ¹⁹⁵ in P such that, for each $i \in \{1,\ldots,m\}$, there exists $x \ge \lfloor \frac{y}{2} \rfloor$ 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 $a^x \in L(C_i)$.

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

- 201 Maj_A(x,y) $\iff \left\lceil \frac{y}{2} \right\rceil \le x \le y \text{ and } a^x \in L(A);$
- 202 Min_A(x,y) $\iff \left\lceil \frac{y}{2} \right\rceil \le x < y \text{ and } a^{y-x} \in L(A);$

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

Let us give and justify a list of Horn clauses whose conjunction ψ'_G defines the predicates Maj_A, Min_A and Sum_{BC}, using the arithmetic predicates of R_{arith} (see Notation 2 and Remark 10), namely $R^{x=y}, R^{y=2x}$ and $R^{\rho(x,y)}$, for $\rho(x,y) \coloneqq x \ge \lfloor \frac{y}{2} \rfloor$.

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

$$\texttt{min}(x) \land \texttt{min}(y) \land Q_a(y) \to \texttt{Maj}_A(x,y).$$

Induction on the length y. If we have for y > 1 the inequalities $\left\lceil \frac{y-1}{2} \right\rceil \le x \le y-1$ and $x \ge \left\lceil \frac{y}{2} \right\rceil$ then $\left\lceil \frac{y}{2} \right\rceil \le x \le y$. This justifies the clause:

213 $y > 1 \land \operatorname{Maj}_A(x, y - 1) \land x \ge \left\lceil \frac{y}{2} \right\rceil \to \operatorname{Maj}_A(x, y) \text{ for all } A \in N.$

For y > 1 and y = 2x, we have $a^x = a^{y-x}$ and $\left\lceil \frac{y}{2} \right\rceil \le x < y$. This justifies the clause: $y > 1 \land \operatorname{Maj}_A(x, y - 1) \land y = 2x \to \operatorname{Min}_A(x, y)$ for all $A \in N$.

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

218 $x > 1 \land y > 1 \land \operatorname{Min}_A(x-1,y-1) \to \operatorname{Min}_A(x,y)$ 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:

 $= initialization: \operatorname{Maj}_B(x,y) \wedge \operatorname{Min}_C(x,y) \to \operatorname{Sum}_{\operatorname{BC}}(x,y); \operatorname{Min}_B(x,y) \wedge \operatorname{Maj}_C(x,y) \to \operatorname{Sum}_{\operatorname{BC}}(x,y);$

222 induction: $\neg \min(x) \land \operatorname{Sum}_{\operatorname{BC}}(x-1,y) \to \operatorname{Sum}_{\operatorname{BC}}(x,y).$

Long rules. Each rule $A \to B_1 C_1 \& \dots \& B_m C_m$ of P is expressed by the clause:

 $_{^{224}} \quad \blacksquare \quad x = y \wedge \operatorname{Sum}_{\operatorname{B_1C_1}}(x,y) \wedge \dots \wedge \operatorname{Sum}_{\operatorname{B_mC_m}}(x,y) \to \operatorname{Maj}_A(x,y).$

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

Definition of $a^+ \setminus L$. We have the equivalence $\operatorname{Maj}_S(n,n) \iff a^n \in L(S) \iff a^n \in L$. Therefore, the following contradiction clause expresses $a^n \notin L$:

$$_{229} \quad \blacksquare \quad \gamma_S \coloneqq \max(x) \land \max(y) \land \operatorname{Maj}_S(x,y) \to \bot.$$

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

²³⁴ 3.2 Equivalence of logic with grid-circuits

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

▶ Definition 12. A grid-circuit is a tuple $C := (\Sigma, (Input_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{acc}, \mathbf{g})$ where

²³⁸ Σ is the input alphabet and $(\text{Input}_n)_{n>0}$ is the family of input functions ²³⁹ Input_n: $\Sigma^n \times [1, n]^2 \to \Sigma \cup \{\$\}$ such that, for $w = w_1 \dots w_n \in \Sigma^n$,

Input_n $(w, x, y) = w_y$ if x = 1 and Input_n(w, x, y) =\$ otherwise,

²⁴¹ $\mathbf{Q} \cup \{ \sharp \}$ is the finite set of states and $\mathbf{Q}_{acc} \subseteq \mathbf{Q}$ is the subset of accepting states,

 $g: (\mathbf{Q} \cup \{\sharp\})^2 \times (\Sigma \cup \{\$\}) \to \mathbf{Q}$ is the transition function.

▶ Definition 13 (computation of a grid-circuit). The computation C_w of a grid-circuit $C := (\Sigma, (\text{Input}_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{\text{acc}}, \mathbf{g})$ on $a w = w_1 \dots w_n \in \Sigma^n$ is a regular grid of $(n+1)^2$ sites $(x, y) \in [0, n]^2$, each in a state $\langle x, y \rangle \in \mathbf{Q} \cup \{\sharp\}$ computed inductively:

each site in $\{0\} \times [0, n]$ or $[0, n] \times \{0\}$ is in the particular state \sharp ;

 $\text{ = the state of each site } (x,y) \in [1,n]^2 \text{ is } \langle x,y \rangle = \mathsf{g}(\langle x,y-1 \rangle, \langle x-1,y \rangle, \mathtt{Input}_n(w,x,y)).$



Figure 3 The grid-circuit

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 belongs to \mathbf{Q}_{acc} . The language recognized by C is the set of words it accepts. We denote by Grid the class of languages recognized by a grid-circuit.

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

```
▶ Lemma 14. [10] pred-ESO-HORN = Grid.
```

Proof. In some sense, a grid-circuit is the "normalized form" of a formula of pred-ESO-HORN. So, the inclusion $Grid \subseteq pred$ -ESO-HORN is proved in a straightforward way.

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

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

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

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

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

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

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: $x > 1 \land S(x-1,y) \to S^{x-1}(x,y), x > 1 \land S^{x-1}(x-1,y) \to S^{x-2}(x,y),$
- $_{267} \quad y > 1 \land S^{x-2}(x, y-1) \to S^{x-2, y-1}(x, y), \text{ and } y > 1 \land S^{x-2, y-1}(x, y-1) \to S^{x-2, y-2}(x, y),$

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

It is clear that the formula $\Phi := \exists \mathbf{R} \forall x \forall y \psi$ is equivalent to the formula $\Phi' := \exists \mathbf{R}' \forall x \forall y \psi'$ 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 $\psi_{\text{replace}} \land \psi_{\text{def}}$, 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}' .

Thus, any formula $\Phi \in \text{pred-ESO-HORN}$ is equivalent to a formula $\Phi' \in \text{pred-ESO-HORN}$ whose computation clauses only contain hypotheses of the following three forms:

 $R(x-1,y) \wedge x > 1$; $R(x,y-1) \wedge y > 1$; 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

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

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

The second idea is to "solve" the Horn clauses θ_i according to the input clauses and *all* the possible conjunctions of hypotheses α_i that may be true. Notice the two following facts: the hypotheses of the input clauses are input literals and the conjuncts of the α_i 's are of the 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 x + y that the obtained formula Φ' in which no atom R(x, y) appears as a clause hypothesis, is equivalent to the above formula Φ . The complete proof is given in Appendix A.

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

292 (a) $x = 1 \land y = 1 \land Q_s(y) \to R_j(x, y);$

²⁹³ (b) $x = 1 \land y > 1 \land Q_s(y) \land \bigwedge_{i \in A} R_i(x, y - 1) \to R_j(x, y);$

²⁹⁴ (c) $x > 1 \land y = 1 \land \bigwedge_{i \in A} R_i(x - 1, y) \to R_j(x, y);$

 $^{ _{ 295 }} (\mathrm{d}) \ x > 1 \wedge y > 1 \wedge \bigwedge_{i \in A} R_i(x-1,y) \wedge \bigwedge_{i \in B} R_i(x,y-1) \rightarrow R_j(x,y);$

296 (e) $x = n \land y = n \land R_j(x, y) \to \bot$.

Now, transform this formula into a grid-circuit $C \coloneqq (\Sigma, (\text{Input}_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{\text{acc}}, \mathbf{g})$. The 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 true. Let \mathbf{Q} be the power set of the set of \mathbf{R} indices: $\mathbf{Q} \coloneqq \mathcal{P}([1, m])$. There are four types of transition (a-d) which mimic the clauses (a-d) above. These are, for $s \in \Sigma$ and $q, q' \in \mathbf{Q}$:

(a) $g(\sharp, \sharp, s) = \{j \in [1, m] \mid \text{there is a clause (a) with } Q_s, \text{ and conclusion } R_j(x, y)\};$

(b) $\mathbf{g}(q,\sharp,s) = \{j \in [1,m] \mid \text{there is a clause (b) with } Q_s, \text{ and } A \subseteq q, \text{ and conclusion } R_j(x,y)\};$

(c) $\mathbf{g}(\sharp, q, \$) = \{j \in [1, m] \mid \text{there is a clause (c) with } A \subseteq q, \text{ and conclusion } R_j(x, y)\};$

 $\text{(d) } g(q,q',\$) = \{ j \in [1,m] \mid \exists \text{ a clause (d) with } A \subseteq q, B \subseteq q', \text{ and conclusion } R_j(x,y) \}.$

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

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

309 3.3 Grid-circuits are equivalent to real-time 1-CA

- 310 ► Lemma 15. [10] Grid = 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 pred-ESO-HORN = Grid = RealTimeCA. These equalities trivially hold when restricted ³¹⁵ to unary languages: pred-ESO-HORN₁ = Grid₁ = RealTimeCA₁.

From the fact that the class RealTimeCA₁ is closed under complement and from Lemma 11, we deduce $Conj_1 \subseteq pred-ESO-HORN_1 = Grid_1 = RealTimeCA_1$.

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

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

Theorem 16. Conj \subseteq RealTime20CA.

³²¹ 4.1 Expressing a conjunctive language in logic: the general case

The generating process of a conjunctive language is naturally expressed in the Horn logic incl-pred-ESO-HORN. This is a hybrid logic with three first-order variables x, y, z, whose name means that it makes inductions on the variable interval [x, y], by *inclusion*, and on the 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).

▶ Definition 17 (incl-pred-ESO-HORN). A formula of incl-pred-ESO-HORN is a formula 326

- $\Phi \coloneqq \exists \mathbf{R} \forall x \forall y \forall z \psi(x, y, z) \text{ where } \mathbf{R} \text{ is a finite set of ternary predicates, and } \psi \text{ is a conjunction}$ 327
- of Horn clauses, of signature $S_{\Sigma} \cup \mathbf{R} \cup \{=, \leq\}$, and of the three following forms: 328
- an input clause: $x = y \wedge \min(z) \wedge Q_s(x) \rightarrow R(x, y, z)$ with $s \in \Sigma$ and $R \in \mathbf{R}$; 329
- a computation clause: $\delta_1 \wedge \ldots \wedge \delta_r \to R(x, y, z)$ with $R \in \mathbf{R}$ and where each hypothesis δ_i 330
- 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 331
- $S \in \mathbf{R}$ and $a, b, c \ge 0$ three integers such that a + b + c > 0; 332
- a contradiction clause: $\min(x) \wedge \max(y) \wedge \max(z) \wedge R(x, y, z) \rightarrow \bot$ with $R \in \mathbf{R}$. 333
- We denote by incl-pred-ESO-HORN the class of languages defined by a formula of incl-pred-ESO-HORN. 334

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

Proof. The proof is a variation (an extension) of the proof of the same result, Lemma 11, in 336 the unary case. This is why we insist on the differences. Let $G = (\Sigma, N, P, S)$ be a conjunctive 337 grammar in binary normal form which generates L and let w be a word $w = w_1 \dots w_n \in \Sigma^+$. 338 For each $A \in N$ and each factor $w_{x,y} \coloneqq w_x \dots w_y$, we have, according to the length y - x + 1339 of $w_{x,y}$, the following equivalences which will be the basis of our induction: 340

if x = y, then $w_{x,y} \in L(A) \iff$ the short rule $A \to w_x$ belongs to P; 341

if x < y, then $w_{x,y} \in L(A) \iff$ there is a long rule $A \rightarrow B_1 C_1 \& \dots \& B_m C_m$ 342 in P such that, for each $i \in \{1, \ldots, m\}$, there exists $z \ge \lceil (y - x + 1)/2 \rceil$ such that 343 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 $w_{y-z+1,y} \in L(C_i)$. 344 Thus, a double induction is performed, on the index interval [x, y] of a factor $w_{x,y}$ and 345 on the maximal z among the lengths of the two sub-factors u, v of the m decompositions 346 $w_{x,y} = uv, u \in L(B_i), v \in L(C_i)$, for a long rule. This is naturally expressed in the logic 347 incl-pred-ESO-HORN. 348

We want to construct a first-order formula $\forall x \forall y \forall z \psi_G$ of signature $S_{\Sigma} \cup \mathbf{R} \cup \{=, \leq\}$, 349 for the set of *ternary* predicates $\mathbf{R} \coloneqq \{ \mathtt{Pref}_A^{\mathtt{Maj}}, \mathtt{Pref}_A^{\mathtt{Maj}}, \mathtt{Suff}_A^{\mathtt{Maj}}, \mathtt{Suff}_A^{\mathtt{Maj}} \mid A \in N \} \cup$ 350 $\{\text{Concat}_{BC} \mid B, C \in N\} \cup \mathbf{R}_{\text{arith}}$, so that the formula $\Phi_G \coloneqq \exists \mathbf{R} \forall x \forall y \forall z \psi_G$ belongs to 351 incl-pred-ESO-HORN and defines the language $\Sigma^+ \setminus L$. The intuitive meanings of the predic-352 ates $\operatorname{Pref}_{A}^{\operatorname{Maj}}$, $\operatorname{Pref}_{A}^{\operatorname{Min}}$, $\operatorname{Suff}_{A}^{\operatorname{Maj}}$, $\operatorname{Suff}_{A}^{\operatorname{Min}}$ and $\operatorname{Concat}_{BC}$ are as follows: $= \operatorname{Pref}_{A}^{\operatorname{Maj}}(x, y, z) \iff \begin{bmatrix} \frac{y-x+1}{2} \end{bmatrix} \le z \le y - x + 1 \text{ and } w_{x,x+z-1} \in L(A);$ $= \operatorname{Pref}_{A}^{\operatorname{Min}}(x, y, z) \iff \begin{bmatrix} \frac{y-x+1}{2} \end{bmatrix} \le z \le y - x \text{ and } w_{x,y-z} \in L(A);$ 353

- 354
- 355

356 **Suff**_A^{Maj}(x, y, z)
$$\iff \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le y-x+1 \text{ and } w_{y-z+1,y} \in L(A);$$

357

 $\begin{array}{c|c} & \operatorname{Suff}_A^{\operatorname{Min}}(x,y,z) \iff \left\lceil \frac{y-\bar{x}+1}{2} \right\rceil \leq z \leq y-x \text{ and } w_{x+z,y} \in L(A); \\ & \operatorname{Concat}_{BC}(x,y,z) \iff \text{ there is some } z' \text{ with } \left\lceil \frac{y-x+1}{2} \right\rceil \leq z' \leq z \text{ such that } \end{array}$ 358 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 $w_{y-z'+1,y} \in L(C)$. 359

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

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

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

 $= x = y \wedge z = 1 \wedge Q_s(x) \to \operatorname{Pref}_A^{\operatorname{Maj}}(x, y, z) \; ; \; x = y \wedge z = 1 \wedge Q_s(x) \to \operatorname{Suff}_A^{\operatorname{Maj}}(x, y, z).$ 366

- **Induction for prefixes.** If we have for x < y the inequalities 367
- $\left\lceil \frac{(y-1)-x+1}{2} \right\rceil \le z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \text{ then } \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le y-x+1. \text{ This } z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le y-x+1. \text{ This } z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le y-x+1. \text{ This } z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le y-x+1. \text{ This } z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le y-x+1. \text{ This } z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le y-x+1. \text{ This } z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le (y-1)-x+1. \text{ This } z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le (y-1)-x+1. \text{ This } z \le (y-1)-x+1 \text{ and } z \ge \left\lceil \frac{y-x+1}{2} \right\rceil \le z \le (y-1)-x+1. \text{ This } z \le (y-1)-x+1. \text{$ 368 justifies the clause: 369
- $x \leq y 1 \wedge \operatorname{Pref}_{A}^{\operatorname{Maj}}(x, y 1, z) \wedge z \geq \left\lceil \frac{y x + 1}{2} \right\rceil \to \operatorname{Pref}_{A}^{\operatorname{Maj}}(x, y, z), \text{ for all } A \in N.$ 370

For x < y and y - x + 1 = 2z, we have $w_{x,x+z-1} = w_{x,y-z}$ and $\left\lfloor \frac{y-x+1}{2} \right\rfloor \leq z \leq y-x$. 371 This justifies the clause: 372 $x \leq y - 1 \wedge \operatorname{Pref}_{A}^{\operatorname{Maj}}(x, y - 1, z) \wedge y - x + 1 = 2z \to \operatorname{Pref}_{A}^{\operatorname{Min}}(x, y, z), \text{ for all } A \in N.$ 373 For x < y and z > 1 and $\left\lceil \frac{(y-1)-x+1}{2} \right\rceil \le z-1 \le (y-1)-x$, we have $\left\lceil \frac{y-x+1}{2} \right\rceil \le z \le y-x$. 374 This justifies the clause: 375 $x \leq y - 1 \land z > 1 \land \operatorname{Pref}_{A}^{\operatorname{Min}}(x, y - 1, z - 1) \to \operatorname{Pref}_{A}^{\operatorname{Min}}(x, y, z), \text{ for all } A \in N.$ 376 Induction for suffixes. As this induction is symmetric to the one for prefixes, we do not 377 justify the following list of induction clauses for the predicates $\text{Suff}_{A}^{\text{Maj}}$ and $\text{Suff}_{A}^{\text{Min}}$, $A \in N$: 378 $\begin{array}{l} \textbf{x}+1 \leq y \wedge \texttt{Suff}_A^{\texttt{Maj}}(x+1,y,z) \wedge z \geq \left\lceil \frac{y-x+1}{2} \right\rceil \rightarrow \texttt{Suff}_A^{\texttt{Maj}}(x,y,z); \\ \textbf{x}+1 \leq y \wedge \texttt{Suff}_A^{\texttt{Maj}}(x+1,y,z) \wedge y - x + 1 = 2z \rightarrow \texttt{Suff}_A^{\texttt{Min}}(x,y,z); \\ \textbf{x}+1 \leq y \wedge z > 1 \wedge \texttt{Suff}_A^{\texttt{Min}}(x+1,y,z-1) \rightarrow \texttt{Suff}_A^{\texttt{Min}}(x,y,z). \end{array}$ 379 380 381 **Concatenation.** For all $B, C \in N$, it is clear that the concatenation predicate $Concat_{BC}$ 382 is defined inductively by the following three clauses: 383 $\begin{array}{l} \hline \quad initialization: \mbox{Pref}_B^{\tt Maj}(x,y,z) \land \mbox{Suff}_C^{\tt Min}(x,y,z) \rightarrow \mbox{Concat}_{BC}(x,y,z); \\ \mbox{Pref}_B^{\tt Min}(x,y,z) \land \mbox{Suff}_C^{\tt Maj}(x,y,z) \rightarrow \mbox{Concat}_{BC}(x,y,z); \end{array}$ 384 385 induction: $z > 1 \land \texttt{Concat}_{BC}(x, y, z - 1) \rightarrow \texttt{Concat}_{BC}(x, y, z)$. 386 **Long rules.** Each rule $A \to B_1 C_1 \& \dots \& B_m C_m$ of P is expressed by the two clauses: 387 $= z = y - x + 1 \wedge \texttt{Concat}_{B_1C_1}(x, y, z) \wedge \dots \wedge \texttt{Concat}_{B_mC_m}(x, y, z) \rightarrow \texttt{Pref}_A^{\texttt{Maj}}(x, y, z);$ 388 $= z = y - x + 1 \wedge \texttt{Concat}_{B_1C_1}(x, y, z) \wedge \dots \wedge \texttt{Concat}_{B_mC_m}(x, y, z) \rightarrow \texttt{Suff}_A^{\texttt{Maj}}(x, y, z).$ 389 Thus, the formula $\forall x \forall y \forall z \psi'_G$ where ψ'_G is the conjunction of the above clauses defines the 390 predicates $\operatorname{Pref}_{A}^{\operatorname{Maj}}, \operatorname{Pref}_{A}^{\operatorname{Min}}, \operatorname{Suff}_{A}^{\operatorname{Maj}}, \operatorname{Suff}_{A}^{\operatorname{Min}}, \operatorname{and} \operatorname{Concat}_{BC}$. 391 **Definition of** $\Sigma^+ \setminus L$. We have the equivalence $\operatorname{Pref}_S^{\operatorname{Maj}}(1, n, n) \iff w \in L(S) \iff w \in L$. 392 Therefore, the following contradiction clause expresses $w \notin L$: 393 394 Finally, observe that the formula $\Phi_G := \exists \mathbf{R} \forall x \forall y \forall z \psi_G$ where ψ_G is $\gamma_{\text{arith}} \land \psi'_G \land \gamma_S$

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

³⁹⁹ 4.2 Equivalence of logic with cube-circuits

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

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

⁴⁰³ = Σ is the input alphabet and $(\text{Input}_n)_{n>0}$ is the family of input functions ⁴⁰⁴ Input_n: $\Sigma^n \times [1,n]^3 \to \Sigma \cup \{\$\}$ such that, for $w = w_1 \dots w_n \in \Sigma^n$,

Input_n $(w, x, y, z) = w_y$ if x = y and z = 1, and Input_n(w, x, y, z) =\$ otherwise,

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

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

⁴⁰⁸ ► **Definition 20** (computation of a cube-circuit). The computation C_w of a cube-circuit ⁴⁰⁹ $C := (\Sigma, (\text{Input}_n)_{n>0}, \mathbf{Q}, \mathbf{Q}_{acc}, \mathbf{g})$ 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 $\langle x, y, z \rangle \in \mathbf{Q} \cup \{\sharp\}$ computed inductively: ⁴¹¹ = each site (x, y, z) such that x > y or z = 0 is in the state \sharp ;

XX:12 Conjunctive grammars, cellular automata and logic

the state of each site
$$(x, y, z) \in [1, n]^3$$
 such that $x \leq y$ and $z > 0$ is

 $_{^{413}} \qquad \langle x,y\rangle = \mathsf{g}(\langle x+1,y,z\rangle,\langle x,y-1,z\rangle,\langle x,y,z-1\rangle,\mathtt{Input}_n(w,x,y,z)).$

A word $w = w_1 \dots w_n \in \Sigma^n$ is accepted by the cube-circuit C if the output state (1, n, n)

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.

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

425 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 = RealTime20CA.

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

438

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

From the fact that the class RealTime20CA is closed under complement and from Lemma 18, we deduce $Conj \subseteq incl-pred-ESO-HORN = Cube = RealTime20CA.$



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

443 **5** Conclusion

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



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

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

456 — References

- Serge Abiteboul, Richard Hull, and Victor Vianu. Foundations of Databases. Addison-Wesley, 1995.
- ⁴⁵⁹ 2 Nicolas Bacquey, Etienne Grandjean, and Frédéric Olive. Definability by Horn Formulas and 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.*,
- 68(1):19-36, 1989.
- Marianne Delorme and Jacques Mazoyer. Cellular Automata as Language Recognizers in Cellular Automata: a Parallel Model. Kluwer, 1999.
- 6 Ronald Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In
 Complexity of Computation, SIAM-AMS Proceedings, pages 43–73, 1974.
- Toora Giammarresi, Antonio Restivo, Sebastian Seibert, and Wolfgang Thomas. Monadic
 second-order logic over rectangular pictures and recognizability by tiling systems. *Inf. Comput.*,
 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.
- Etienne Grandjean and Théo Grente. Descriptive complexity for minimal time of cellular
 automata. In *LICS*, 2019, pages 1–13, 2019.
- Etienne Grandjean, Théo Grente, and Véronique Terrier. Inductive definitions in logic versus
 programs of real-time cellular automata. *hal.archives-ouvertes.fr/hal-02474520/ submitted to Theoretical Computer Science*, 62 pages, February 2020.
- Etienne Grandjean and Frédéric Olive. A logical approach to locality in pictures languages.
 Journal of Computer and System Science, 82(6):959–1006, 2016.
- A85 13 Oscar H. Ibarra and Tao Jiang. Relating the power of cellular arrays to their closure properties.
 Theor. Comput. Sci., 57:225–238, 1988.
- 487 14 Neil Immerman. Descriptive complexity. Springer, 1999.
- Artur Jez. Conjunctive grammars generate non-regular unary languages. Int. J. Found.
 Comput. Sci., 19(3):597-615, 2008.
- 490 16 K. N. King. Alternating multihead finite automata. Theor. Comput. Sci., 61:149–174, 1988.
- Hans Kleine Büning and Theodor Lettmann. Propositional logic deduction and algorithms,
 volume 48 of Cambridge tracts in theoretical computer science. Cambridge University Press,
 1999.
- I8 S. Rao Kosaraju. Speed of recognition of context-free languages by array automata. SIAM J.
 Comput., 4(3):331–340, 1975.
- Leonid Libkin. *Elements of Finite Model Theory*. Texts in Theoretical Computer Science. An
 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.
- Alexander Okhotin. On the equivalence of linear conjunctive grammars and trellis automata.
 Theoretical Informatics and Applications, 38(1):69–88, 2004.
- Alexander Okhotin. Conjunctive and boolean grammars: The true general case of the
 context-free grammars. *Computer Science Review*, 9:27–59, 2013.
- Véronique Terrier. Closure properties of cellular automata. Theor. Comput. Sci., 352(1-3):97–107, 2006.
- Véronique Terrier. Low complexity classes of multidimensional cellular automata. Theor.
 Comput. Sci., 369(1-3):142–156, 2006.
- Véronique Terrier. Language recognition by cellular automata. In *Handbook of Natural Computing*, pages 123–158. Springer, 2012.
- ⁵¹⁰ 27 Hao Wang. Dominoes and the aea case of the decision problem. In *Proceedings on the Symposium on the Mathematical Theory of Automata, April 1962*, pages 23–55, 1963.

523

512 Appendix A: Complement of proof for Lemma 14

Elimination of hypotheses R(x, y): 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. Accordingly, the formula obtained Φ can be rewritten in the form

⁵¹⁶
$$\Phi \coloneqq \exists \mathbf{R} \forall x \forall y \left[\bigwedge_{i} C_{i}(x,y) \land \bigwedge_{i \in [1,k]} (\alpha_{i}(x,y) \to \theta_{i}(x,y)) \right]$$

where the C_i 's are the input clauses and the contradiction clause and each computation 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 of the only forms $R(x-1, y) \land \neg \min(x)$, $R(x, y-1) \land \neg \min(y)$ (but not R(x, y)), and $\theta_i(x, y)$ is a Horn clause whose all atoms are of the form R(x, y).

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

$$K_J \coloneqq \{h \in [1,m] \mid \bigwedge_{i \in J} \theta_i(x,y) \to R_h(x,y) \text{ is a tautology} \}.$$

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

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
$$\Phi' \coloneqq \exists \mathbf{R} \forall x \forall y \left[\bigwedge_{i} C_i(x, y) \land \bigwedge_{J \subseteq [1, k]} \bigwedge_{h \in K_J} \left(\bigwedge_{i \in J} \alpha_i(x, y) \to R_h(x, y) \right) \right]$$

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

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

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

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

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

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

Lemma 24 (Horn property: folklore). Let F be a strict Horn formula of propositional calculus, that is a conjunction of clauses of the form $p_1 \land \ldots \land p_k \to p_0$ where $k \ge 0$ and the

XX:16 Conjunctive grammars, cellular automata and logic

- p_i 's are propositional variables. Let F' be the conjunction of propositional variables q such
- that the implication $F \rightarrow q$ is a tautology. F has the same minimal model ⁵ as F'.

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 model of the Horn formula

⁵⁴⁷
$$\varphi' \coloneqq \forall x \forall y \left[\bigwedge_{i} C_i(x,y) \land \bigwedge_{J \subseteq [1,k]} \bigwedge_{h \in K_J} \left(\bigwedge_{i \in J} \alpha_i(x,y) \to R_h(x,y) \right) \right].$$

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

$${}_{^{549}} \qquad \varphi \coloneqq \forall x \forall y \left[\bigwedge_{i} C_i(x,y) \land \bigwedge_{i \in [1,k]} (\alpha_i(x,y) \to \theta_i(x,y)) \right].$$

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

$$\varphi_t \coloneqq \forall x \forall y \ \left[x + y \le t \to \left[\bigwedge_i C_i(x, y) \land \bigwedge_{i \in [1, k]} (\alpha_i(x, y) \to \theta_i(x, y)) \right] \right]$$

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

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

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

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

Recall that each $\alpha_i(a, b)$ is a (possibly empty) conjunction of atoms R(a', b') with (a', b') = (a - 1, b) or (a', b') = (a, b - 1), therefore such that a' + b' = t - 1. Let $J \subseteq [1, k]$ be any set. 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 $\bigwedge_{h \in K_J} (\bigwedge_{i \in J} \alpha_i(a, b) \to R_h(a, b))$ 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 $\bigwedge_{h \in K_J} (\bigwedge_{i \in J} \alpha_i(a,b) \to R_h(a,b))$ holds in $(\langle w \rangle, \mathbf{R})$.

From (1) and (2), we deduce that in $(\langle w \rangle, \mathbf{R})$ the conjunction $\bigwedge_{J \subseteq [1,k]} \bigwedge_{h \in K_J} (\bigwedge_{i \in J} \alpha_i(a,b) \rightarrow R_h(a,b))$ is equivalent to the conjunction $\bigwedge_{J \subseteq J_{a,b}} \bigwedge_{h \in K_J} R_h(a,b)$, which can be simplified 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 $h \in [1,m]$, the minimal model $(\langle w \rangle, \mathbf{R})$ of the Horn formula φ' satisfies the atom $R_h(a,b)$ iff h belongs 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 $I(p_2) = I(p_4) = 0$.

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

577 or, equivalently,

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

579 As a consequence of Lemma 24, the two conjunctions

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

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

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

Appendix B: Complement of proof for Lemma 15

Grid ⊆ RealTimeCA. To prove this inclusion, we show how to simulate the computation of
 the grid-circuit on a real-time CA. The simulation is made by a geometric transformation that
 embeds the grid-circuit in the space-time diagram of a real-time CA. This transformation is
 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' < 1⁵⁹⁶ is send 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.

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

576

578

580