Modanese, Augusto

Sublinear-Time Probabilistic Cellular Automata

Published in:
40th International Symposium on Theoretical Aspects of Computer Science, STACS 2023

DOI:
10.4230/LIPIcs.STACS.2023.47

Published: 01/03/2023

Document Version
Publisher's PDF, also known as Version of record

Sublinear-Time Probabilistic Cellular Automata

Augusto Modanese ✉
Aalto University, Espoo, Finland

Abstract

We propose and investigate a probabilistic model of sublinear-time one-dimensional cellular automata. In particular, we modify the model of ACA (which are cellular automata that accept if and only if all cells simultaneously accept) so that every cell changes its state not only dependent on the states it sees in its neighborhood but also on an unbiased coin toss of its own. The resulting model is dubbed *probabilistic ACA* (PACA). We consider one- and two-sided error versions of the model (in the same spirit as the classes RP and BPP) and establish a separation between the classes of languages they can recognize all the way up to $o(\sqrt{n})$ time. As a consequence, we have a $\Omega(\sqrt{n})$ lower bound for derandomizing constant-time one-sided error PACAs (using deterministic ACAs). We also prove that derandomization of $T(n)$-time PACAs (to polynomial-time deterministic cellular automata) for various regimes of $T(n) = \omega(\log n)$ implies non-trivial derandomization results for the class RP (e.g., $\mathsf{P} = \mathsf{RP}$). The main contribution is an almost full characterization of the constant-time PACA classes: For one-sided error, the class equals that of the deterministic model; that is, constant-time one-sided error PACAs can be fully derandomized with only a constant multiplicative overhead in time complexity. As for two-sided error, we identify a natural class we call the *linearly testable languages* (LLT) and prove that the languages decidable by constant-time two-sided error PACAs are "sandwiched" in-between the closure of LLT under union and intersection and the class of locally threshold testable languages (LTT).

2012 ACM Subject Classification Theory of computation → Formal languages and automata theory

Keywords and phrases Cellular automata, local computation, probabilistic models, subregular language classes

Digital Object Identifier 10.4230/LIPIcs.STACS.2023.47

Related Version *Full Version*: `https://arxiv.org/abs/2203.14614` [15]

Funding Supported by Helsinki Institute for Information Technology (HIIT). Much of this work done while affiliated with the Karlsruhe Institute of Technology (KIT).

Acknowledgements I would like to thank Thomas Worsch for the helpful discussions and feedback as well as the anonymous reviewers for their insightful comments.

1 Introduction

Cellular automata (CAs) have been extensively studied as a natural model of distributed computation. A one-dimensional CA is composed of a row of fairly limited computational agents – the *cells* – which, by interacting with their immediate neighbors, realize a global behavior and work towards a common goal.

As every model of computation, CAs have been widely studied as language acceptors [10, 20]. These efforts apparently were almost exclusively devoted to the linear- or real-time case – to the detriment of the *sublinear-time* one [14]. This is unfortunate since, as it was recently shown in [13], the study of sublinear-time CA variants might help better direct efforts in resolving outstanding problems in computational complexity theory.

In this work, we consider a *probabilistic* sublinear-time CA model. Our main goal is to analyze to what extent – if at all – the addition of randomness to the model is able to make up for its inherent limitations. (For instance, sublinear-time CA models are usually restricted to a local view of their input [14] and are also unable to cope with long unary subwords [13].)


1.1 The Model

We consider only bounded one-dimensional cellular automata.

▶ **Definition 1** (Cellular automaton). *A* cellular automaton *(CA) is a triple* $C = (Q, \$, \delta)$ *where* Q *is the finite set of* states, $\$ \notin Q$ *is the* boundary symbol, *and* $\delta\colon Q_\$ \times Q \times Q_\$ \to Q$ *is the* local transition function, *where* $Q_\$ = Q \cup \{\$\}$. *The elements in the domain of* δ *are the possible* local configurations *of the cells of* C. *For a fixed width* $n \in \mathbb{N}_+$, *the* global configurations *of* C *are the elements of* Q^n. *The cells* 0 *and* $n-1$ *are the* border cells *of* C. *The* global transition function $\Delta\colon Q^n \to Q^n$ *is obtained by simultaneous application of* δ *everywhere; that is, if* $s \in Q^n$ *is the current global configuration of* C, *then*

$$\Delta(s) = \delta(\$, s_0, s_1)\,\delta(s_0, s_1, s_2)\cdots\delta(s_{n-2}, s_{n-1}, \$).$$

For $t \in \mathbb{N}_0$, Δ^t *denotes the* t*-th iterate of* Δ. *For an initial configuration* $s \in Q^n$, *the sequence* $s = \Delta^0(s), \Delta(s), \Delta^2(s), \ldots$ *is the* orbit *of* C *(for* s*). Writing the orbit of* C *line for line yields its* space-time diagram.

One key theme connecting CAs and models of physics is *causality*: If two cells i and j are t cells away from each other, then j requires at least t steps to receive any information from i. In the *sublinear-time* case, this means every cell only gets to see a very small section of the input. In some sense this is reminiscent of *locality* in circuits (e.g., [22]), though locality in the CA model carries a more literal meaning since it is connected to the notion of space (whereas in circuits there is no equivalent notion). One should keep this limitation (of every cell only seeing a portion of the input) in mind as it is central to several of our arguments.

The usual acceptance condition for CA-based language recognizers is that of a distinguished cell (usually the leftmost one) entering an accepting state [10]. This is unsuitable for sublinear-time computation since then the automaton is limited to verifying prefixes of a constant length [14]. The most widely studied [8, 9, 14, 18] acceptance condition for sublinear-time is that of all cells simultaneously accepting, yielding the model of *ACA* (where the first "A" in the acronym indicates that *all* cells must accept).

▶ **Definition 2** (DACA). *A* deterministic ACA *(DACA) is a CA* C *with an* input alphabet $\Sigma \subseteq Q$ *as well as a subset* $A \subseteq Q$ *of* accepting states. *We say* C accepts *an input* $x \in \Sigma^+$ *if there is* $t \in \mathbb{N}_0$ *such that* $\Delta^t(x) \in A^n$, *and we denote the set of all such* x *by* $L(C)$. *In addition,* C *is said to have* time complexity *(bounded by)* $T\colon \mathbb{N}_+ \to \mathbb{N}_0$ *if, for every* $x \in L(C) \cap \Sigma^n$, *there is* $t < T(|x|)$ *such that* $\Delta^t(x) \in A^n$.

We propose a probabilistic version of the ACA model inspired by the stochastic automata of [2] and the definition of probabilistic Turing machines (see, e.g., [1]). In the model of *probabilistic ACA* (PACA), at every step, each cell tosses a fair coin $c \in \{0, 1\}$ and then changes its state based on the outcome of c. There is a nice interplay between this form of randomness access and the overall theme of *locality* in CAs: Random events pertaining to a cell i depend exclusively on what occurs in the vicinity of i. Furthermore, events corresponding to distinct cells i and j can only be dependent if i and j are near each other; otherwise, they are necessarily independent (see Lemma 11).

We consider both one- and two-sided error versions of the model as natural counterparts of RP and BPP machines, respectively. Although PACAs are a conceptually simple extension of ACAs, the definition requires certain care, in particular regarding the model's time complexity. We refer to Section 3 for the formal definitions and further discussion.

Finally, we should also mention our model is more restricted than a *stochastic CA*,[1] which is a CA in which the next state of a cell is chosen according to an arbitrary distribution that depends on the cell's local configuration. For a survey on stochastic CAs, we refer to [11].

1.2 Results

Inclusion relations. As can be expected, two-sided error PACAs are more powerful than their one-sided error counterparts. Say a DACA C is *equivalent* to a PACA C' if they accept the same language (i.e., $L(C) = L(C')$).

▶ **Theorem 3.** *The following hold:*

1. *If C is a one-sided error PACA with time complexity T, then there is an equivalent two-sided error PACA C' with time complexity $O(T)$.*
2. *There is a language L recognizable by constant-time two-sided error PACA but not by any $o(\sqrt{n})$-time one-sided error PACA.*

We stress the first item does not follow immediately from the definitions since it requires error reduction by a constant factor, which requires a non-trivial construction. It remains open whether in the second item we can improve the separation from $o(\sqrt{n})$ to $o(n)$ time. Nevertheless, as it stands the result already implies a lower bound of $\Omega(\sqrt{n})$ time for the derandomization (to a DACA) of constant-time two-sided error PACA.

Another result we show is how time-efficient derandomization of PACA classes imply derandomization results for RP (with a trade-off between the PACA time complexity and the efficiency of the derandomization).

▶ **Theorem 4.** *Let $d \geq 1$. The following hold:*

- *If there is $\varepsilon > 0$ such that every n^ε-time (one- or two-sided error) PACA can be converted into an equivalent n^d-time deterministic CA, then $\mathsf{P} = \mathsf{RP}$.*
- *If every $\mathsf{polylog}(n)$-time PACA can be converted into an equivalent n^d-time deterministic CA, then, for every $\varepsilon > 0$, $\mathsf{RP} \subseteq \mathsf{TIME}[2^{n^\varepsilon}]$.*
- *If there is $b > 2$ so that any $(\log n)^b$-time PACA can be converted into an equivalent n^d-time deterministic CA, then, for every $a \geq 1$ and $c > a/(b-1)$, $\mathsf{RTIME}[n^a] \subseteq \mathsf{TIME}[2^{O(n^c)}]$.*

We write "deterministic CA" instead of "DACA" since, for $T(n) = \Omega(n)$, a T-time DACA is equivalent to an $O(T)$-time deterministic CA with the usual acceptance condition [14].

Characterization of constant time. As a first step we analyze and almost completely characterize constant-time PACA. Indeed, the constant-time case is already very rich and worth considering in and of itself. This may not come as a surprise since other local computational models (e.g., local graph algorithms [19]) also exhibit behavior in the constant-time case that is far from trivial.

In Appendix A we give an example of a one-sided error PACA that recognizes a language L strictly faster than any DACA for L. Nonetheless, as we prove, one-sided error PACA can be derandomized with only a constant multiplicative overhead in time complexity.

▶ **Theorem 5.** *For any constant-time one-sided error PACA C, there is a constant-time DACA C' such that $L(C) = L(C')$.*

[1] Unfortunately, the literature uses the terms stochastic and probabilistic CA interchangeably. We deem "probabilistic" more suitable since it is intended as a CA version of a probabilistic Turing machine.

In turn, the class of languages accepted by constant-time two-sided error PACA can be considerably narrowed down in terms of a novel subregular class LLT, dubbed the *locally linearly testable* languages. Below, $\mathsf{LLT}_{\cup\cap}$ is the closure over LLT under union and intersection and LTT its Boolean closure (i.e., its closure under union, intersection, and complement).

▶ **Theorem 6.** *The class of languages that can be accepted by a constant-time two-sided error PACA contains* $\mathsf{LLT}_{\cup\cap}$ *and is strictly contained in* LTT*.*

It is known that the constant-time class of DACA equals the closure under union $\mathsf{SLT}_{\cup}$ of the strictly local languages SLT [18]. (We refer to Section 4.2 for the definitions.) Since $\mathsf{SLT}_{\cup} \subsetneq \mathsf{LLT}_{\cup}$ is a proper inclusion, this gives a separation of the deterministic and probabilistic classes in the case of two-sided error and starkly contrasts with Theorem 5.

As far as we are aware of, the class LLT does not previously appear in the literature.[2] In Section 4.2 we show LLT lies in-between $\mathsf{SLT}_{\cup}$ and the class of locally threshold testable languages LTT. In this regard LLT is similar to the class LT of locally testable languages; however, as we can also show, both LLT and $\mathsf{LLT}_{\cup}$ are incomparable to LT.

1.3 Organization

The rest of the paper is organized as follows: Section 2 introduces basic concepts and notation. Following that, in Section 3 we define the PACA model and prove standard error reduction results as well as Theorem 3. In Section 4 we focus on the constant-time case and prove Theorems 5 and 6. Finally, in Section 5 we address the general sublinear-time case and prove Theorem 4. We conclude with Section 6 by mentioning a few further research directions.

2 Preliminaries

It is assumed the reader is familiar with the theory of cellular automata as well as with basic notions of computational complexity theory (see, e.g., the standard references [1, 4, 6]).

All logarithms are to the base 2. The set of integers is denoted by $\mathbb{Z}$, that of non-negative integers by $\mathbb{N}_0$, and that of positive integers by $\mathbb{N}_+$. For a set S and $n, m \in \mathbb{N}_+$, $S^{n\times m}$ is the set of n-row, m-column matrices over S. For $n \in \mathbb{N}_+$, $[n] = \{i \in \mathbb{N}_0 \mid i < n\}$ is the set of the first n non-negative integers. Also, for $a, b \in \mathbb{Z}$, by $[a, b] = \{i \in \mathbb{Z} \mid a \leq i \leq b\}$ we always refer to an interval containing only integers.

Symbols in words are indexed starting with zero. The i-th symbol of a word w is denoted by w_i. For an alphabet Σ and $n \in \mathbb{N}_0$, $\Sigma^{\leq n}$ contains the words $w \in \Sigma^*$ for which $|w| \leq n$. For an infix $m \in \Sigma^{\leq |w|}$ of w, $|w|_m$ is the number of occurrences of m in w. Without restriction, the empty word is not an element of any language that we consider. (This is needed for definitional reasons; see Definitions 1 and 2 below.)

We write U_n (resp., $U_{n\times m}$) for a random variable distributed uniformly over $\{0,1\}^n$ (resp., $\{0,1\}^{n\times m}$). We also need the following variant of the Chernoff bound (see, e.g., [21]):

▶ **Theorem 7** (Chernoff bound)**.** *Let* $X_1, \ldots, X_n$ *be independently and identically distributed Bernoulli variables and* $\mu = \mathrm{E}[X_i]$*. There is a constant* $c > 0$ *such that the following holds for every* $\varepsilon = \varepsilon(n) > 0$*:*

$$\Pr\left[\left|\frac{\sum_i X_i}{n} - \mu\right| > \varepsilon\right] < 2^{-cn\varepsilon^2}.$$

[2] A similar class is perhaps found in [5, 17], but there are clear distinctions to be made between the two.

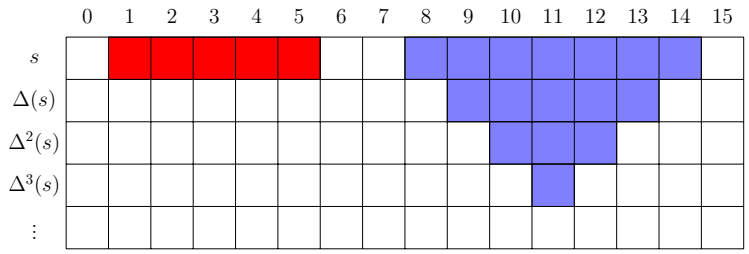

Figure 1 Space-time diagram of a CA with 16 cells for an initial configuration s. (States have been omitted for simplicity.) The cells marked in red form the 2-neighborhood of cell 3, the ones in blue the 3-lightcone of cell number 11.

Many of our low-level arguments make use of the notion of a *lightcone*.[3] For a set S and non-negative integers $n \leq m$, a lightcone $L = (\ell_{i,j})$ of *radius* m and *height* n over S is a trapezoidal (when $n < m$) or triangular (when $n = m$) array of elements $\ell_{i,j} \in S$, where $i \in [0, n]$ and $j \in [-m, m]$:

$$\begin{array}{ccccccccccc} \ell_{0,-m} & \ell_{0,-m+1} & \cdots & & \cdots & \cdots & \ell_{0,0} & \cdots & \cdots & \cdots & \ell_{0,m-1} & \ell_{0,m} \\ & \ell_{1,-m+1} & \cdots & & \cdots & \cdots & \ell_{1,0} & \cdots & \cdots & \cdots & \ell_{1,m-1} & \\ & & \ddots & & & & \vdots & & & \iddots & & \\ & & & \ell_{n,-m+n} & & \cdots & \ell_{n,0} & \cdots & \ell_{n,m-n} & & & \end{array}$$

The element $\ell_{0,0}$ is the *center* of the lightcone. The *layers* of L are indexed by i, where the i-th layer contains $2(m-i)+1$ elements. Hence, the top layer contains $2m+1$ elements and the bottom one $2(m-n)+1$; in particular, the bottom layer is a single element if and only if $n = m$. There are $\sum_{i=0}^{n}(2(m-i)+1) = (n+1)(2m-n+1)$ elements in a lightcone in total.

▶ **Definition 8** (Neighborhoods and lightcones). *Let C be a CA and $n \in \mathbb{N}_+$. For $i \in [n]$ and $r \in \mathbb{N}_0$, the interval $[i-r, i+r] \cap [n]$ forms the* r-neighborhood *of i. For $t \in \mathbb{N}_0$, the* t-lightcone *of i is the lightcone of radius and height t centered at i in the 0-th row (i.e., the initial configuration) of the space-time diagram of C.*[4]

3 Fundamentals

As customary for randomized models of computation, one may consider both online and offline views of our model. Since it gives a more natural presentation, in the definition below we first assume an online perspective and then address the definitional issue mentioned in the introduction. In the last part, we switch to an offline view that we will use for the rest of the paper; this is more comfortable to work with since we can then refer to the cells' coin tosses explicitly.

▶ **Definition 9** (PACA). *Let Q be a finite set of states and $\Sigma \subseteq Q$ an alphabet. A* probabilistic ACA *(PACA) C is a CA with two local transition functions $\delta_0, \delta_1 \colon Q^3 \to Q$. At each step of C, each cell tosses a fair coin $c \in \{0,1\}$ and updates its state according to δ_c; that is, if the current configuration of C is $s \in Q^n$ and the cells obtain coin tosses $r = r_0 \cdots r_{n-1} \in \{0,1\}^n$ (where r_i is the coin toss of the i-th cell), then the next configuration of C is*

[3] Some sources distinguish between *future* and *past* lightcones. Here we shall need only past lightcones.
[4] If the lightcone's dimensions overstep the boundaries of the space-time diagram (i.e., i is too close to either of the borders of C (e.g., $i < t$)), then some cells in the t-lightcone will have undefined states. In this case, we set the undefined states to \$, which ensures consistency with δ.

$$\Delta_r(s) = \delta_{r_0}(\$, s_0, s_1)\,\delta_{r_1}(s_0, s_1, s_2)\cdots\delta_{r_{n-1}}(s_{n-2}, s_{n-1}, \$).$$

Seeing this process as a Markov chain M over Q^n, we recast the global transition function $\Delta = \Delta_{U_n}$ as a family of random variables $(\Delta(s))_{s\in Q^n}$ parameterized by the current configuration s of C, where $\Delta(s)$ is sampled by starting in state s and performing a single transition on M (having drawn the cells' coin tosses according to U_n). Similarly, for $t \in \mathbb{N}_0$, $\Delta^t(s)$ is sampled by starting in s and performing t transitions on M.

A computation *of C for an input $x \in \Sigma^n$ is a path in M starting at x. The computation is* accepting *if the path visits A^n at least once. In order to be able to quantify the probability of a PACA accepting an input, we additionally require for every PACA C that there is a function $T\colon \mathbb{N}_+ \to \mathbb{N}_0$ such that, for any input $x \in \Sigma^n$, every accepting computation for x visits A^n for the first time in strictly less than $T(n)$ steps; that is, if there is $t \in \mathbb{N}_0$ with $\Delta^t(x) \in A^n$, then $\Delta^{t_1}(x) \in A^n$ for some $t_1 < T(n)$. (Hence, every accepting computation for x has an initial segment with endpoint in A^n and whose length is strictly less than $T(n)$.) If this is the case for any such T, then we say C has* time complexity *(bounded by) T.*

With this condition in place, we may now equivalently replace the coin tosses of C with a matrix $R \in \{0,1\}^{T(n)\times n}$ of bits with rows $R_0, \ldots, R_{T(n)-1}$ and such that $R_j(i)$ corresponds to the coin toss of the i-th cell in step j. (If C accepts in step t, then the coin tosses in rows $t, \ldots, T(n)-1$ are ignored.) We refer to R as a random input *to C.*[5] *Blurring the distinction between the two perspectives (i.e., online and offline randomness), we write $C(x, R) = 1$ if C accepts x when its coin tosses are set according to R, or $C(x, R) = 0$ otherwise.*

Definition 9 states the acceptance condition for a *single* computation (i.e., one fixed choice of a random input); however, we must still define acceptance based on *all* computations (i.e., for random inputs picked according to a uniform distribution). The two most natural candidates are the analogues of the well-studied classes RP and BPP, which we define next.

▶ **Definition 10** (*p*-error PACA). *Let $L \subseteq \Sigma^*$ and $p \in [0, 1)$. A* one-sided p-error PACA for L *is a PACA C with time complexity $T = T(n)$ such that, for every $x \in \Sigma^n$,*

$$x \in L \iff \Pr[C(x, U_{T\times n}) = 1] \geq 1 - p \quad \text{and} \quad x \notin L \iff \Pr[C(x, U_{T\times n}) = 1] = 0.$$

If $p = 1/2$, then we simply say C is a one-sided error PACA. *Similarly, for $p < 1/2$, a* two-sided p-error PACA for L *is a PACA C with time complexity $T = T(n)$ for which*

$$x \in L \iff \Pr[C(x, U_{T\times n}) = 1] \geq 1 - p \quad \text{and} \quad x \notin L \iff \Pr[C(x, U_{T\times n}) = 1] \leq p$$

hold for every $x \in \Sigma^*$. If $p = 1/3$, then we simply say C is a two-sided error PACA. *In both cases, we write $L(C) = L$ and say C* accepts *L.*

Note that, to each 0-error PACA C, one can obtain an equivalent DACA C' with the same time complexity by setting the local transition function to δ_0. In the rest of the paper, if it is not specified which of the two variants above (i.e., one- or two-sided error) is meant, then we mean both variants collectively.

[5] The number of rows of R is dependent on the choice of T. This is not an issue here since any superficial rows are ignored by C; that is, without restriction we may take T to be such that every value $T(n)$ is minimal and set the number of rows of R to $T(n)$. The motivation for letting R be larger is that, when simulating a PACA, it may be the case that it is more convenient (or even possible) to compute only an upper bound $T'(n) \geq T(n)$ instead of the actual minimal value $T(n)$.

From the perspective of complexity theory, it is interesting to compare the PACA model with probabilistic circuits. It is known that every $T(n)$-time DACA can be simulated by an L-uniform AC circuit (i.e., a Boolean circuit with gates of unbounded fan-in) having $\mathsf{poly}(n)$ size and $O(\max\{1, T(n)/\log n\})$ depth [14]. Using the same approach as in [14], we note the same holds for PACAs if we use probabilistic AC circuits instead. The proof in [14] bases on descriptive complexity theory, the central observation being that the state of a cell i after $\log n$ steps given its $(\log n)$-neighborhood is a predicate that is computable in logarithmic space. Hence, for the PACA case we need only factor in the auxiliary random input into this predicate.

A key property that PACAs have but probabilistic circuits do not, however, is *distance* between computational units. (Indeed, in circuits, there is no such thing as the "length" of a wire.) One consequence of this is the following simple fact.

▶ **Lemma 11** (Independence of local events). *Let C be a one- or two-sided error PACA, let $x \in \Sigma^n$ be an input to C, and let $T \in \mathbb{N}_+$. In addition, let $i, j \in [n]$ be such that $|i - j| > 2(T-1)$ and E_i (resp., E_j) be an event described exclusively by the states of the i-th (resp., j-th) cell of C in the time steps $0, \dots, T-1$ (e.g., the i-th cell accepts in some step t where $t < T$). Then E_i and E_j are independent.*

Proof. For any random input R, the states of $k \in \{i, j\}$ in the time steps between 0 and $T-1$ is uniquely determined by the values of $R(t, k-T+t+1), \dots, R(t, k+T-t-1)$ for $t \in [T]$. Without loss of generality, suppose $i \leq j$. Since $i + T - 1 < j - T + 1$, E_i and E_j are conditioned on disjoint sets of values of R, thus implying independence. ◀

Note the proof still holds in case $T = 1$, in which case the events E_i and E_j occur with probability either 0 or 1, thus also (trivially) implying independence.

3.1 Robustness of the Definition

We now prove that the definition of PACA is robust with respect to the choice of $p = 1/2$ (resp., $p = 1/3$) for the error of one-sided (resp., two-sided) error PACA.

For one-sided error, we can reduce the error p to any desired constant value p'.

▶ **Proposition 12.** *Let $p, p' \in (0, 1)$ be constant and $p' < p$. For every one-sided p-error PACA C, there is a one-sided p'-error PACA C' such that $L(C) = L(C')$. Furthermore, if C has time complexity $T(n)$, then C' has time complexity $O(T(n))$.*

It follows that the definition of PACA is robust under the choice of p (as long as it is constant) and regardless of the time complexity (up to constant multiplicative factors).

The proof is essentially a generalization of the idea used in [14] to show that the sublinear-time DACA classes are closed under union. Namely, C' simulates several copies $C_0, \dots, C_{m-1}$ of C in parallel and accepting if and only if at least one C_i accepts. This idea is particularly elegant because m can be chosen to be constant and we update the C_i in a round-robin fashion (i.e., first C_0, then C_1, C_2, etc., and finally C_0 again after C_{m-1}). The alternative is to simulate each C_i for $T(n)$ steps at a time, which is not possible in general since we would have to compute $T(n)$ first. The construction we give avoids this issue entirely.

Proof. We construct a PACA C' with the desired properties. Let $m = \lceil \log(1/p' - 1/p) \rceil$. Furthermore, let Q be the state set of C and Σ its input alphabet. We set the state set of C' to $Q^m \times [m] \cup \Sigma$. Given an input x, every cell of C' initially changes its state from $x(i)$ to $(x(i), \dots, x(i), 0)$. The cells of C' simulate m copies of C as follows: If the last component of

a cell contains the value j, then its j-th component[6] q_0 is updated to $\delta(q_{-1}, q_0, q_1)$, where q_{-1} and q_1 are the j-th components of the left and right neighbors, respectively (or \$ in case of a border cell); at the same time, the last component of the cell is set to $j+1$ if $j < m$ or 0 in case $j = m$. A cell of C' is accepting if and only if its last component is equal to j and its j-th component is an accepting state of C.

Denote the i-th simulated copy of C by C_i. Clearly, C' accepts in step $mt + i + 1$ for $i \in [m]$ if and only if C_i accepts in step t, so we immediately have that C' has $O(T(n))$ time complexity. For the same reason and since C' never accepts in step 0, C' does not accept any input $x \notin L(C)$. As for $x \in L(C)$, note the m copies of C are all simulated using independent coin tosses, thus implying

$$\Pr[C' \text{ does not accept } x] = \Pr[\forall i \in [m] : C_i \text{ does not accept } x] < p^m \leq p'.$$

Hence, C' accepts x with probability at least $1 - p'$, as desired. ◀

For two-sided error, we show the same holds for every choice of p for *constant-time* PACA. We remark the construction is considerably more complex than in the one-sided error case.

▶ **Proposition 13.** *Let $p, p' \in (0, 1/2)$ be constant and $p' < p$. For every two-sided p-error PACA C with constant time complexity $T = O(1)$, there is a two-sided p'-error PACA C' with time complexity $O(T) = O(1)$ and such that $L(C) = L(C')$.*

It remains open whether a similar result holds for general (i.e., non-constant-time) two-sided error PACA. Generalizing our proof of Proposition 13 would require at the very least a construction for intersecting non-constant-time PACA languages. Note that closure under intersection is open in the deterministic setting (i.e., of DACA) as well [14].

3.2 One- vs. Two-Sided Error

The results of Section 3.1 are also useful in obtaining the following:

▶ **Theorem 3.** *The following hold:*

1. *If C is a one-sided error PACA with time complexity T, then there is an equivalent two-sided error PACA C' with time complexity $O(T)$.*
2. *There is a language L recognizable by constant-time two-sided error PACA but not by any $o(\sqrt{n})$-time one-sided error PACA.*

Proof. The first item follows from Proposition 12: Transform C into a one-sided error PACA C' with error at most $1/3$ and then notice that C' also qualifies as a two-sided error PACA (as it simply never errs on "no" instances). For the second item, consider the language

$$L = \{x \in \{0,1\}^+ \mid |x|_1 \leq 1\}.$$

We obtain a constant-time two-sided error PACA for L as follows: If a cell receives a 0 as input, then it immediately accepts; otherwise, it collects two random bits r_0 and r_1 in the first two steps and then, seeing $r_0 r_1$ as the binary representation of an integer $1 \leq t \leq 4$, it accepts (only) in the subsequent t-th step. Hence, if the input x is such that $|x|_1 \leq 1$, the PACA always accepts; conversely, if $|x|_1 \geq 2$, then the PACA only accepts if all 1 cells pick the same value for t, which occurs with probability at most $1/4$.

[6] In the same manner as we do for the indices of a word, we number the components starting with zero.

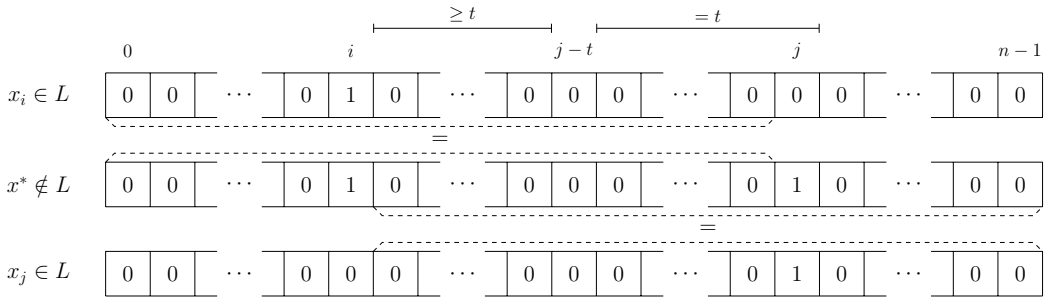

Figure 2 Constructing $x^* \notin L$ from $x_i, x_j \in L$. The numbers above the cells indicate their respective indices. Since every t-neighborhood of x^* appears in either x_i or x_j and both x_i and x_j are accepted in (exactly) t steps, it follows that C accepts x^* in t steps.

It remains to show $L(C) \neq L$ for any T-time one-sided error PACA C where $T = o(\sqrt{n})$. Let n be large enough so that $T = T(n) \leq \sqrt{n}/2$. Observe that $L \cap \{0,1\}^n = \{0^n, x_1, \dots, x_n\}$ where $x_i = 0^{i-1}10^{n-i}$. Let us now assume that $x_i \in L(C)$ holds for every i. Since C accepts with probability at least $1/2$, by the pigeonhole principle there is R such that $C(x_i, R) = 1$ for at least a $1/2$ fraction of the x_i. In addition, by averaging there is a step $t < T$ such that at least a $1/2T$ fraction of the x_i is accepted by C in step t. Since there are $n/2T \geq 2T \geq 2t+2$ such x_i, we can find $i, j \in [n]$ with $j \geq i + 2t + 1$ and $x_i, x_j \in L(C)$. Consider now the input

$$x^* = 0^{i-1}10^{j-i-1}10^{n-j},$$

which is not in L. We argue $C(x^*, R) = 1$, thus implying $L(C) \neq L$ and completing the proof.

We can see this by comparing the local views of the "bad" word x^* with the "good" ones x_i and x_j (see Figure 2): Let $k \in [n]$ be any cell of C. If $k < j - t$, then the t-neighborhood of k on input x^* is identical to that when the input is x_i, so k must be accepting in step t. Similarly, if $k \geq j - t$, then the t-neighborhood of k in x^* is the same as in x_j, so k is accepting as well. It follows all cells of C are accepting in step t for inputs x^* and R. ◀

4 The Constant-Time Case

In this section we now focus on constant-time PACA. Our goal will be to characterize the constant-time classes of both one- and two-sided error PACA (i.e., Theorems 5 and 6). First, we introduce the concept of *critical cells*, which is central to our analysis.

▶ **Definition 14** (Critical cell). *Let C be a one- or two-sided error PACA, and let $x \in L(C) \cap \Sigma^n$. We say a cell $i \in [n]$ is* critical *for x in step $t \in \mathbb{N}_0$ if there are random inputs $R, R' \in \{0,1\}^{t \times n}$ such that i is accepting in step t of $C(x, R)$ but not in step t of $C(x, R')$.*

In other words, if E is the event of i being accepting in step t of C on input x, then $0 < \Pr[E] < 1$ (where the probability is taken over the coin tosses of C). We should stress that whether a cell is critical or not may be *highly dependent* on x and t; for instance, there may be inputs $x_1 \neq x_2$ where the cell i is critical for x_1 but not for x_2.

As it turns out, the number of critical cells of a constant-time PACA is also constant.

▶ **Lemma 15.** *Let C be a T-time (one- or two-sided error) PACA for $T \in \mathbb{N}_+$, and let $x \in L(C) \cap \Sigma^n$. In addition, let $t \in [T]$ be a time step in which x is accepted by C with non-zero probability. Then there are $2^{O(T^2)}$ cells that are critical for x in step t. It follows there are $T \cdot 2^{O(T^2)} = 2^{O(T^2)}$ critical cells for x in total (i.e., all such time steps comprised).*

Proof. For $i \in [n]$, let E_i denote the event in which the i-th cell accepts in step t. Assume towards a contradiction that there is $x \in L(C)$ in which strictly more than $2T \cdot (1+\log T) \cdot 2^{T^2} = 2^{O(T^2)}$ cells are critical for x (in step t). By inspection, this implies there is a set $K \subseteq [n]$ of $|K| \geq (1 + \log T) \cdot 2^{T^2}$ cells such that every $k \in K$ is critical for x and, for every distinct $i, j \in K$, $|i - j| \geq 2T$. (For instance, in the extreme case where every $0 \leq i < 2T \cdot 2^{T^2}$ is critical, pick $K = \{2jT \mid j \in [2^{T^2}]\}$.) Since there are at most T^2 coin tosses that determine whether a cell $i \in K$ accepts or not (i.e., those in the T-lightcone of i), $\Pr[E_i] < 1$ if and only if $\Pr[E_i] \leq 1 - 2^{-T^2}$. By Lemma 11, the events E_i are all independent, implying

$$\Pr[C \text{ accepts } x \text{ in step } t] \leq \prod_{i \in K} \Pr[E_i] \leq \left(1 - \frac{1}{2^{T^2}}\right)^{|K|} < \frac{1}{e^{1+\log T}} < \frac{1}{2T}.$$

Since t was arbitrary, by a union bound it follows that the probability that C accepts x in step t is strictly less than $1/2$. This contradicts $x \in L(C)$ both when C is a one- and a two-sided error PACA. ◀

4.1 Characterization of One-Sided Error PACA

▶ **Theorem 5.** *For any constant-time one-sided error PACA C, there is a constant-time DACA C' such that $L(C) = L(C')$.*

Lemma 15 implies that, given any $x \in L(C)$, the decision of C accepting can be traced back to a set K of critical cells where $|K|$ is constant. To illustrate the idea, suppose that, if C accepts, then it always does so in a fixed time step $t < T$; in addition, assume the cells in K are all far apart (e.g., more than $2(T-1)$ cells away from each other as in Lemma 11).

Let us *locally* inspect the space-time diagram of C for x, that is, by looking at the t-lightcone of each cell i. Then we notice that, if $i \in K$, there is a choice of random bits in the t-lightcone of i that causes i to accept; conversely, if $i \notin K$, then *any* setting of random bits results in i accepting. Consider how this changes when $x \notin L(C)$ while assuming K remains unchanged. Every cell $i \notin K$ still behave the same; that is, it accepts regardless of the random input it sees. As for the cells in K, however, since they are all far apart, it cannot be the case that we still find random bits for every $i \in K$ that cause i to accept; otherwise C would accept x with non-zero probability, contradicting the definition of one-sided error PACA. Hence, there must be at least some $i \in K$ that *never* accepts. In summary, (under these assumptions) we can locally distinguish $x \in L(C)$ from $x \notin L(C)$ by looking at the cells of K and checking whether, for every $i \in K$, there is at least one setting of the random bits in the t-lightcone of i that causes it to accept.

To obtain the proof, we must generalize this idea to handle the case where the cells in K are not necessarily far from each other – which in particular means we can no longer assume that the events of them accepting are independent – as well as of K varying with the input. Since Lemma 15 only gives an upper bound for critical cells when the input is in $L(C)$, we must also account for the case where the input is not in $L(C)$ and $|K|$ exceeds said bound.

Proof. Let $T \in \mathbb{N}_+$ be the time complexity of C, and let M be the upper bound on the number of critical cells (for inputs in $L(C)$) from Lemma 15. Without restriction, we may assume $T > 1$. We first give the construction for C' and then prove its correctness.

Construction. Given an input $x \in \Sigma^n$, the automaton C' operates in two phases. In the first one, the cells communicate so that, in the end, each cell is aware of the inputs in its r-neighborhood, where $r = (2M-1)(T-1)$. (Note this is possible because r is constant.) The

second phase proceeds in T steps, with the cells assuming accepting states or not depending on a condition we shall describe next. After the second phase is over (and C' has not yet accepted), all cells unconditionally enter a non-accepting state and maintain it indefinitely. Hence, C' only ever accepts during the second phase.

We now describe when a cell $i \in [n]$ is accepting in the t-th step of the second phase, where $t \in [T]$. Let K be the set of critical cells of x in step t. The decision process is as follows:

1. First the cell checks whether there are strictly more than M cells in its r-neighborhood N that are critical in step t of C. If i cannot determine this for any cell $j \in N$ (since it did not receive the entire t-neighborhood of j during the first phase), then j is simply ignored. If this is the case, then i assumes a non-accepting state (in the t-th step of the second phase).
2. The cell then determines whether it is critical itself in step t of C. If this is *not* the case, then it becomes accepting if and only if it is accepting in step t of C (regardless of the random input).
3. Otherwise i is critical in step t. Let $B_i \subseteq [n]$ be the subset of cells that results from the following sequence of operations:
   a. Initialize B_i to $\{i\}$.
   b. For every cell $j \in B_i$, add to B_i every $k \in K$ such that $|j - k| \leq 2(T-1)$.
   c. Repeat step 2 until a fixpoint is reached.

   (This necessarily terminates since there are at most M critical cells in N and we checked the upper bound of M previously.) By choice of r, we have that $|i - j| \leq 2(M-1)(T-1)$ for every $j \in B_i$. In particular, we have that the t-neighborhood of every $j \in B_i$ is completely contained in N, which means cell i is capable of determining B_i.[7] The cell i then accepts if and only if there is a setting of random bits in the lightcone L_i of radius r and height t centered at i that causes every cell in B_i to accept in step t of C.

This process repeats itself in every step t of the second phase. Note that it can be performed by i instantaneously (i.e., without requiring any additional time steps of C') since it can be hardcoded into the local transition function δ.

Correctness. It is evident that C' is a constant-time PACA, so all that remains is to verify its correctness. To that end, fix an input x and consider the two cases:

$x \in L(C)$. Then there is a random input R such that C accepts x in step $t \in [T]$. This means that, for every critical cell $i \in K$, if we set the random bits in L_i according to R, then every cell in B_i accepts in step t of C. Likewise, every cell $i \notin K$ is accepting in step t of C by definition. In both cases we have that i also accepts in the t-th step of the second phase of C', thus implying $x \in L(C')$.

$x \notin L(C)$. Then, for every random input R and every step $t \in [T]$, there is at least one cell $i \in [n]$ that is not accepting in the t-th step of $C(x, R)$. If i is not critical, then i is also not accepting in the t-th step of the second phase of C' (regardless of the random input), and thus C' also does not accept x. Hence, assume that every such i (i.e., every i such that there is a random input R for which i is not accepting in the t-th step of $C(x, R)$) is a critical cell.

[7] Note it is not necessary for i to be aware of the actual numbers of the cells in B_i; it suffices for it to compute their positions relative to itself. For example, if $i = 5$ and $B_i = \{4, 5\}$, then it suffices for i to regard $j = 4$ as cell -1 (relative to itself). Hence, by "determining B_i" here we mean that i computes only these relative positions (and not the absolute ones, which would be impossible to achieve in only constant time).

Let $J \subseteq [n]$ denote the set of all such cells and, for $i \in J$, let $D_i \subseteq J$ be the subset that contains every $j \in J$ such that the events of i and j accepting in step t are *not* independent (conditioned on the random input to C). In addition, let A_i denote the event in which every cell of D_i is accepting in step t of $C(x, U_{T\times n})$. We show the following:

▷ Claim. There is an $i \in J$ such that $\Pr[A_i] = 0$; that is, for every R, there is at least one cell in D_i that is not accepting in step t of $C(x, R)$.

This will complete the proof since then i is also not accepting in the t-th step of the second phase of C' (since any cell in D_i is necessarily at most $2(M-1)(T-1)$ cells away from i), thus implying $x \notin L(C')$.
To see the claim is true, suppose towards a contradiction that, for every $i \in J$, we have $\Pr[A_i] > 0$. If there are $i, j \in J$ such that $j \notin D_i$ (and similarly $i \notin D_j$), then by definition

$$\Pr[C(x, U_{T\times n}) = 1] \geq \Pr[A_i \wedge A_j] = \Pr[A_i]\Pr[A_j] > 0,$$

contradicting $x \notin L(C)$. Thus, there must be $i \in J$ such that $J = D_i$; however, this then implies

$$\Pr[C(x, U_{T\times n}) = 1] = \Pr[A_i] > 0.$$

As this is also a contradiction, the claim (and hence the theorem) follows. ◀

4.2 Characterization of Two-Sided Error PACA

We introduce some notation. For $\ell \in \mathbb{N}_0$ and a word $w \in \Sigma^*$, $p_\ell(w)$ is the prefix of w of length ℓ if $|w| \geq \ell$, or w otherwise; similarly, $s_\ell(w)$ is the suffix of length ℓ if $|w| \geq \ell$, or w otherwise. The set of infixes of w of length (exactly) ℓ is denoted by $I_\ell(w)$.

The subregular language classes from the next definition are due to McNaughton and Papert [12] and Beauquier and Pin [3].

▶ **Definition 16** (SLT, LT, LTT). *A language $L \subseteq \Sigma^*$ is* strictly locally testable *if there is $\ell \in \mathbb{N}_+$ and sets $\pi, \sigma \subseteq \Sigma^{\leq \ell}$ and $\mu \subseteq \Sigma^\ell$ such that, for every $w \in \Sigma^*$, $w \in L$ if and only if $p_{\ell-1}(w) \in \pi$, $I_\ell(w) \subseteq \mu$, and $s_{\ell-1}(w) \in \sigma$. The class of all such languages is denoted by* SLT.

A language $L \subseteq \Sigma^*$ is locally testable *if there is $\ell \in \mathbb{N}_+$ such that, for every $w_1, w_2 \in \Sigma^*$ with $p_{\ell-1}(w_1) = p_{\ell-1}(w_2)$, $I_\ell(w_1) = I_\ell(w_2)$, and $s_{\ell-1}(w_1) = s_{\ell-1}(w_2)$, we have that $w_1 \in L$ if and only if $w_2 \in L$. The class of locally testable languages is denoted by* LT.

A language $L \subseteq \Sigma^*$ is locally threshold testable *if there are $\theta, \ell \in \mathbb{N}_+$ such that, for any two words $w_1, w_2 \in \Sigma^*$ for which the following conditions hold, $w_1 \in L$ if and only if $w_2 \in L$:*
1. *$p_{\ell-1}(w_1) = p_{\ell-1}(w_2)$ and $s_{\ell-1}(w_1) = s_{\ell-1}(w_2)$.*
2. *For every $m \in \Sigma^\ell$, if $|w_i|_m < \theta$ for any $i \in \{1, 2\}$, then $|w_1|_m = |w_2|_m$.*

The class of locally threshold testable languages is denoted by LTT.

The class LT equals the closure of SLT under Boolean operations (i.e., union, intersection, and complement) and the inclusion $\mathsf{SLT} \subsetneq \mathsf{LT}$ is proper. As for LTT, it is well-known that it contains every language $\mathsf{Th}(m, \theta) = \{w \in \Sigma^* \mid |w|_m \leq \theta\}$ where $m \in \Sigma^*$ and $\theta \in \mathbb{N}_0$. Also, we have that $\mathsf{LT} \subsetneq \mathsf{LTT}$ and that LTT is closed under Boolean operations. We write $\mathsf{SLT}_\cup$ for the closure of SLT.

▶ **Definition 17** (LLT). *For $\ell \in \mathbb{N}_0$, $\theta \in \mathbb{R}_0^+$, $\pi, \sigma \subseteq \Sigma^{\leq \ell-1}$, and $\alpha \colon \Sigma^\ell \to \mathbb{R}_0^+$, $\mathsf{LLin}_\ell(\pi, \sigma, \alpha, \theta)$ denotes the language of all $w \in \Sigma^+$ that satisfy $p_{\ell-1}(w) \in \pi$, $s_{\ell-1}(w) \in \sigma$, and*

$$\sum_{m \in \Sigma^\ell} \alpha(m) \cdot |w|_m \leq \theta.$$

A language $L \subseteq \Sigma^+$ is said to be locally linearly testable *if there are ℓ, π, σ, α, and θ as above such that $L = \mathsf{LLin}_\ell(\pi, \sigma, \alpha, \theta)$. We denote the class of all such languages by* LLT.

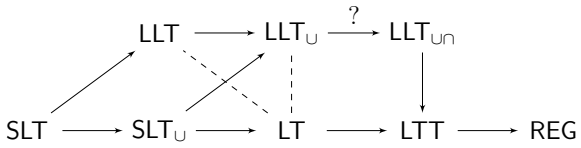

Figure 3 Placement of LLT in the subregular hierarchy. Arrows indicate inclusion relations; all are known to be strict except for the marked one. Dashed lines connect incomparable classes.

We write $\mathsf{LLT}_\cup$ for the closure of LLT under union and $\mathsf{LLT}_{\cup\cap}$ for its closure under both union and intersection.

▶ **Proposition 18.** *The following hold (see Figure 3):*
1. $\mathsf{SLT} \subsetneq \mathsf{LLT} \subsetneq \mathsf{LLT}_\cup \subseteq \mathsf{LLT}_{\cup\cap} \subsetneq \mathsf{LTT}$ *and* $\mathsf{SLT}_\cup \subsetneq \mathsf{LLT}_\cup$.
2. LLT *and* LT *as well as* $\mathsf{LLT}_\cup$ *and* LT *are incomparable.*
3. LTT *equals the Boolean closure of* LLT.

With this terminology in place, we now turn to:

▶ **Theorem 6.** *The class of languages that can be accepted by a constant-time two-sided error PACA contains* $\mathsf{LLT}_{\cup\cap}$ *and is strictly contained in* LTT.

The theorem is proven by showing the two inclusions. In the following, we give a brief overview of the ideas involved; for the full proof, see Appendix B.

First inclusion. The first step is showing that we can "tweak" the components of the LLT condition so that it is more amenable to being tested by a PACA. In particular, we prove we can assume the $\alpha(m)$ weights are such that $2^{-\alpha(m)}$ can be represented using a constant number of bits. Having done so, the construction is more or less straightforward: We collect the subwords of length ℓ in every cell and then accept with the "correct" probabilities; that is, if a cell sees a subword m, then it accepts with probability $2^{-\alpha(m)}$. To lift the inclusion from LLT to $\mathsf{LLT}_{\cup\cap}$, we show the following result:

▶ **Proposition 19.** *Let C_1 and C_2 be constant-time two-sided error PACA. Then there are constant-time two-sided error PACA $C_\cup$ and $C_\cap$ such that $L(C_\cup) = L(C_1) \cup L(C_2)$ and $C_\cap = L(C_1) \cap L(C_2)$.*

Second inclusion. The second inclusion is considerably more complex. Let C be a PACA with time complexity at most T. The proof again bases on the class LLT and uses the fact that LTT equals the Boolean closure over LLT (which we show as a separate result):
1. As a warm-up, we consider the case where the cells of C accept all independently from one another. In addition, we assume that, if C accepts, then it does so in a fixed time step $t < T$. The argument is then relatively straightforward since we need only consider subwords of length $\ell = 2T + 1$ and set their LLT weight according to the acceptance probability that the cell in the middle of the subword would have in C.
2. Next we relax the requirement on independence between the cells (but still assume a fixed time step for acceptance). The situation then requires quite a bit of care since the LLT condition does not account for subword overlaps at all. For instance, there may be cells c_1 and c_2 that are further than T cells apart and that both accept with non-zero probability but where c_i accepts if and only if c_{3-i} does not (see Appendix A for a concrete example). We solve this issue by blowing up ℓ so that a subword covers not only a single cell's

neighborhood but that of an *entire group* of cells whose behavior may be correlated with one other. Here we once more resort to Lemmas 11 and 15 to upper-bound the size of this neighborhood by a constant.

3. Finally, we generalize what we have shown so that it also holds in the case where C may accept in any step $t < T$. This is the only part in the proof where closure under complement is required. The argument bases on generalizing the ideas of the previous step to the case where the automaton may accept in multiple time steps and then applying the inclusion-exclusion principle.

5 The General Sublinear-Time Case

Recall we say a DACA C is *equivalent* to a PACA C' if $L(C) = L(C')$. In this section, we recall and briefly discuss:

▶ **Theorem 4.** *Let $d \geq 1$. The following hold:*

- *If there is $\varepsilon > 0$ such that every n^ε-time (one- or two-sided error) PACA can be converted into an equivalent n^d-time deterministic CA, then $\mathsf{P} = \mathsf{RP}$.*
- *If every $\mathsf{polylog}(n)$-time PACA can be converted into an equivalent n^d-time deterministic CA, then, for every $\varepsilon > 0$, $\mathsf{RP} \subseteq \mathsf{TIME}[2^{n^\varepsilon}]$.*
- *If there is $b > 2$ so that any $(\log n)^b$-time PACA can be converted into an equivalent n^d-time deterministic CA, then, for every $a \geq 1$ and $c > a/(b-1)$, $\mathsf{RTIME}[n^a] \subseteq \mathsf{TIME}[2^{O(n^c)}]$.*

To obtain Theorem 4, we prove the following more general result:

▶ **Proposition 20.** *There is a constant $c > 0$ such that the following holds: Let monotone functions $T, T', h, p \colon \mathbb{N}_+ \to \mathbb{N}_+$ be given with $h(n) \leq 2^n$, $p(n) = \mathsf{poly}(n)$, and $p(n) \geq n$ and such that, for every n and $p'(n) = \Theta(p(n) \log h(n))$, $T(6h(n)p(n)) \geq cp'(n)$. In addition, for n given in unary, let the binary representation of $h(n)$ and $p'(n)$ be computable in $O(p'(n))$ time by a Turing machine and, for N given in unary, let $T'(N)$ be computable in $O(T'(N))$ time by a Turing machine. Furthermore, suppose that, for every T-time one-sided error PACA C, there is a T'-time DACA C' such that $L(C) = L(C')$. Then*

$$\mathsf{RTIME}[p(n)] \subseteq \mathsf{TIME}[h(n) \cdot T'(h(n) \cdot \mathsf{poly}(n)) \cdot \mathsf{poly}(n)].$$

The first item of Theorem 4 is obtained by setting (say) $T(n) = n^\varepsilon$, $T'(n) = n^d$, and $h(n) = p(n)^{2(1/\varepsilon - 1)}$ (assuming $\varepsilon < 1$). For the second one, letting $p(n) = n^a$ where $a > 0$ is arbitrary and (again) $T'(n) = n^d$, set $T(n) = (\log n)^{2+a/\varepsilon}$ and $h(n) = 2^{n^\varepsilon/(d+1)}$. Finally, for the last one, letting again $p(n) = n^a$ and $T'(n) = n^d$, set $T(n) = (\log n)^b$ and $h(n) = 2^{n^c}$.

At the core of the proof of Proposition 20 is a padding argument. Nevertheless, we cannot stress enough that the padding itself is highly nontrivial. In particular, it requires a clever implementation that ensures it can be verified *in parallel* and also *without initial knowledge of the input length*. To see why this is so, observe that, if we simply use a "standard" form of padding where we map $x \in \{0,1\}^+$ to $x' = x0^{p(|x|)}$ (where $p \colon \mathbb{N}_+ \to \mathbb{N}_0$ gives the desired padding length), then it is impossible for the automaton to distinguish between this and, say, $x'' = x0^{p(|x|)/2}$ in $o(p)$ time (assuming, e.g., $p(|x|) = \Omega(|x|)$). The reason for this is that, since cells are initially completely unaware of their position in the input, the cells with an all-zeroes neighborhood must behave exactly the same. More specifically, we can use an argument as in the proof of Theorem 3 to show that the automaton must behave the same on both x' and x'' (in the sense that it accepts the one if and only if it accepts the other) unless it "looks at the whole input" (i.e., unless it has $\Omega(p(|x|)$ time complexity).

The padding technique we use can be traced back to [8]. In a nutshell, we split the input into blocks of the same size that redundantly encode the input length in a locally verifiable way. More importantly, the blocks are numbered from left to right in ascending order, which also allows us to verify that we have the number of blocks that we need. This is crucial in order to ensure the input is "long enough" and the PACA achieves the time complexity that we desire (as a function of the input length).

Proof. Let $L \in \mathsf{RP}$ be decided by an RP machine R whose running time is upper-bounded by p. Without restriction, we assume $p(n) \geq n$. Using standard error reduction in RP, there is then an RP machine R' with running time $p'(n) = \Theta(p(n) \log h(n))$ (i.e., polynomial in n), space complexity at most $p(n)$, and which errs on $x \in L$ with probability strictly less than $1/2h(n)$. Based on L we define the language

$$L' = \{\mathrm{bin}_n(0)\#x_0\#0^{p(n)}\% \cdots \% \,\mathrm{bin}_n(h(n)-1)\#x_{h(n)-1}\#0^{p(n)} \mid n \in \mathbb{N}_+, x_i \in L \cap \Sigma^n\},$$

where $\mathrm{bin}_n(i)$ denotes the n-bit representation of $i < 2^n$. Note the length of an instance of L' is $N \leq 6h(n)p(n) = O(h(n)\mathsf{poly}(n))$.

We claim there is $c > 0$ such that L' can be accepted in at most $cp'(n) = O(p'(n))$ (and in particular less than $T(N)$) time by a one-sided error PACA C. The construction is relatively straightforward: We refer to each group of cells $\mathrm{bin}_n(i)\#x_i\#0^{p'(n)}$ separated by the % symbols as a *block* and the three binary strings in each block (separated by the # symbols) as its *components*. First each block $a_1\#a_2\#a_3$ checks that its components have correct sizes, that is, that $|a_1| = |a_2|$ and $|a_3| = p(|a_1|)$. Then the block communicates with its right neighbor $b_1\#b_2\#b_3$ (if it exists) and checks that $|a_i| = |b_i|$ for every i and that, if $a_1 = \mathrm{bin}_n(j)$, then $b_1 = \mathrm{bin}_n(j+1)$. In addition, the leftmost block checks that its first component is equal to $\mathrm{bin}_n(0)$; similarly, the rightmost block computes $h(n)$ (in $O(p'(n))$ time) and checks that its first component is equal to $\mathrm{bin}_n(h(n)-1)$. Following these initial checks, each block then simulates R' (using bits from its random input as needed) on the input given in its second component using its third component as the tape. If R' accepts, then all cells in the respective block turn accepting. In addition, the delimiter % is always accepting unless it is a border cell.

Clearly C accepts if and only if its input is correctly formatted and R' accepts every one of the x_i (conditioned on the coin tosses that are chosen for it by the respective cells of C). Using a union bound, the probability that C errs on an input $x \in L'$ is

$$\Pr[C(x, U_{T\times n}) = 0] \leq \sum_{i=0}^{h(n)-1} \Pr[R(x_i) = 0] < \sum_{i=0}^{h(n)-1} \frac{1}{2h(n)} = \frac{1}{2}.$$

In addition, the total running time of C is the time needed for the syntactic checks (requiring $O(p'(n))$ time), plus the time spent simulating R' (again, $O(p'(n))$ time using standard simulation techniques). Hence, we can implement C so that it runs in at most $cp'(n)$ time for some constant $c > 0$, as desired.

Now suppose there is a DACA C' equivalent to C as in the statement of the theorem. We shall show there is a deterministic (single-tape) Turing machine that decides L with the purported time complexity. Consider namely the machine S which, on an input $x \in \{0,1\}^n$ of L, produces the input

$$x' = \mathrm{bin}_n(0)\#x\#0^{p(n)}\% \cdots \% \,\mathrm{bin}_n(h(n)-1)\#x\#0^{p(n)}$$

of L' and then simulates C' on x' for $T'(N)$ steps, accepting if and only if C' does. Producing x' from x requires $O(N \cdot \mathsf{poly}(n))$ time since we need only copy $O(n)$ bits from each block separated by the % delimiters to the next (namely the string x and the number of the

previous block). Using the standard simulation of cellular automata by Turing machines, the subsequent simulation of C' requires $O(N \cdot T'(N))$ time. Checking whether C' accepts or not can be performed in parallel to the simulation and requires no additional time. Hence, the time complexity of S is

$$O(N \cdot \mathsf{poly}(n) + N \cdot T'(N)) = O(h(n) \cdot \mathsf{poly}(n) \cdot T'(h(n) \cdot \mathsf{poly}(n))).$$ ◀

6 Further Directions

LLT and two-sided error PACA. Besides giving a separation between one- and two-sided error, Theorem 6 considerably narrows down the position of the class of languages accepted by constant-time two-sided error PACA in the subregular hierarchy. Nevertheless, even though we now know the class is "sandwiched" in-between $\mathsf{LLT}_{\cup\cap}$ and LTT, we still do not have a precise characterization for it. It is challenging to tighten the inclusion from Theorem 6 because the strategy we follow relies on closure under complement, but (as we also prove) the class of two-sided error PACA is *not* closed under complement. It appears that clarifying the relation between said class and $\mathsf{LLT}_{\cup\cap}$ as well as $\mathsf{LLT}_{\cup\cap}$ itself and $\mathsf{LLT}_{\cup}$ or also LT may give a "hint" on how to proceed.

The general sublinear-time case. Theorem 4 indicates that even polylogarithmic-time PACA can recognize languages for which no deterministic polynomial-time algorithm is currently known. Although the proof of Proposition 20 does yield explicit examples of such languages, they are rather unsatisfactory since in order to accept them we do not need the full capabilities of the PACA model. (In particular, communication between blocks of cells is only required to check certain syntactic properties of the input; once this is done, the blocks operate independently from one another.) It would be very interesting to identify languages where the capabilities of the PACA model are put to more extensive use.

Pseudorandom generators. From the opposite direction, to investigate the limitations of the PACA model, one possibility would be to construct *pseudorandom generators* (PRGs) that fool sublinear-time PACAs. Informally, such a PRG is a function $G\colon \{0,1\}^{s(n)} \to \{0,1\}^{r(n)}$ with $s(n) \ll r(n)$ and having the property that a PACA (under given time constraints) is incapable of distinguishing $G(x)$ from uniform when the seed x is chosen uniformly at random. PRGs have found several applications in complexity theory (see, e.g., [21] for an introduction).

Theorem 4 suggests that an unconditional time-efficient derandomization of PACAs is beyond reach of current techniques, so perhaps *space-efficient* derandomization should be considered instead. Indeed, as a PACA can be simulated by a space-efficient machine (e.g., by adapting the algorithm from [13]), it is possible to recast PRGs that fool space-bounded machines (e.g., [7, 16]) as PRGs that fool PACAs. Nevertheless, we may expect to obtain even better constructions by exploiting the locality of PACAs (which space-bounded machines do not suffer from).

References

1 Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

2 Pablo Arrighi, Nicolas Schabanel, and Guillaume Theyssier. Stochastic cellular automata: Correlations, decidability and simulations. *Fundam. Informaticae*, 126(2-3):121–156, 2013. doi:10.3233/FI-2013-875.

3 Danièle Beauquier and Jean-Eric Pin. Factors of words. In *Automata, Languages and Programming, 16th International Colloquium, ICALP89, Stresa, Italy, July 11-15, 1989, Proceedings*, pages 63–79, 1989. doi:10.1007/BFb0035752.

4 Marianne Delorme and Jacques Mazoyer, editors. *Cellular Automata*. Number 460 in Mathematics and Its Applications. Springer Netherlands, 1999. doi:10.1007/978-94-015-9153-9.

5 Pedro García and José Ruiz. Threshold locally testable languages in strict sense. In Carlos Martín-Vide and Victor Mitrana, editors, *Grammars and Automata for String Processing: From Mathematics and Computer Science to Biology, and Back: Essays in Honour of Gheorghe Paun*, volume 9 of *Topics in Computer Mathematics*, pages 243–252. Taylor and Francis, 2003.

6 Oded Goldreich. *Computational Complexity: A Conceptional Perspective*. Cambridge University Press, 2008.

7 William M. Hoza and David Zuckerman. Simple optimal hitting sets for small-success RL. *SIAM J. Comput.*, 49(4):811–820, 2020. doi:10.1137/19M1268707.

8 Oscar H. Ibarra, Michael A. Palis, and Sam M. Kim. Fast parallel language recognition by cellular automata. *Theor. Comput. Sci.*, 41:231–246, 1985. doi:10.1016/0304-3975(85)90073-8.

9 Sam Kim and Robert McCloskey. A characterization of constant-time cellular automata computation. *Phys. D*, 45(1-3):404–419, 1990. doi:10.1016/0167-2789(90)90198-X.

10 Martin Kutrib. Cellular automata and language theory. In *Encyclopedia of Complexity and Systems Science*, pages 800–823. Springer, 2009. doi:10.1007/978-0-387-30440-3_54.

11 Jean Mairesse and Irène Marcovici. Around probabilistic cellular automata. *Theor. Comput. Sci.*, 559:42–72, 2014. doi:10.1016/j.tcs.2014.09.009.

12 Robert McNaughton and Seymour Papert. *Counter-Free Automata*. The MIT Press, 1971.

13 Augusto Modanese. Lower bounds and hardness magnification for sublinear-time shrinking cellular automata. In Rahul Santhanam and Daniil Musatov, editors, *Computer Science – Theory and Applications – 16th International Computer Science Symposium in Russia, CSR 2021, Sochi, Russia, June 28 – July 2, 2021, Proceedings*, volume 12730 of *Lecture Notes in Computer Science*, pages 296–320. Springer, 2021. doi:10.1007/978-3-030-79416-3_18.

14 Augusto Modanese. Sublinear-time language recognition and decision by one-dimensional cellular automata. *Int. J. Found. Comput. Sci.*, 32(6):713–731, 2021. doi:10.1142/S0129054121420053.

15 Augusto Modanese. Sublinear-time probabilistic cellular automata. *CoRR*, abs/2203.14614, 2022. arXiv:2203.14614.

16 Noam Nisan. Pseudorandom generators for space-bounded computation. *Comb.*, 12(4):449–461, 1992. doi:10.1007/BF01305237.

17 José Ruiz, Salvador España Boquera, and Pedro García. Locally threshold testable languages in strict sense: Application to the inference problem. In *Grammatical Inference, 4th International Colloquium, ICGI-98, Ames, Iowa, USA, July 12-14, 1998, Proceedings*, pages 150–161, 1998. doi:10.1007/BFb0054072.

18 Rudolph Sommerhalder and S. Christian van Westrhenen. Parallel language recognition in constant time by cellular automata. *Acta Inf.*, 19:397–407, 1983. doi:10.1007/BF00290736.

19 Jukka Suomela. Survey of local algorithms. *ACM Comput. Surv.*, 45(2):24:1–24:40, 2013. doi:10.1145/2431211.2431223.

20 Véronique Terrier. Language recognition by cellular automata. In *Handbook of Natural Computing*, pages 123–158. Springer, 2012. doi:10.1007/978-3-540-92910-9_4.

21 Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

22 Andrew Chi-Chih Yao. Circuits and local computation. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*, pages 186–196. ACM, 1989. doi:10.1145/73007.73025.

A An Example for PACA Being More Efficient than DACA

▶ **Example 21.** Let $\Sigma = \{0, 1, 2, 3\}$ and consider the language

$$L = \{0^k 1^l 2^m 3^n \mid k, l, m, n \in \mathbb{N}_0 \text{ and } ((l \geq 2 \text{ and } m \geq 3) \text{ or } (l \geq 3 \text{ and } m \geq 2))\}.$$

No DACA C accepts L in at most 3 steps. This can be shown using methods from [9, 14, 18].

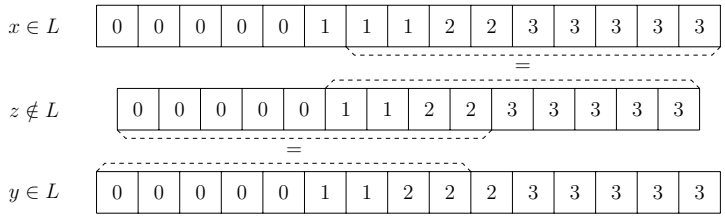

Figure 4 Comparing the words $x = 0^5 1^3 2^2 3^5 \in L$, $y = 0^5 1^2 2^3 3^5 \in L$, and $z = 0^5 1^2 2^2 3^5 \notin L$, we notice that every infix of length 5 of z appears in either x or y. This implies there is no DACA that accepts L with time complexity 3 or less.

Given a DACA C with time complexity 3, we can determine if C accepts a word $x \in \Sigma^*$ by looking only at the infixes of length 5 (and the prefix and suffix of length 4) of x. Consider the words $x = 0^5 1^3 2^2 3^5 \in L$, $y = 0^5 1^2 2^3 3^5 \in L$, and $z = 0^5 1^2 2^2 3^5 \notin L$ (Figure 4). Then every infix of length 5 (and the prefix and suffix of length 4) of z appears in x except for the infixes 00112 and 01122, which both appear in y. It follows that, if $x, y \in L(C)$, then C must also accept z, which proves there is no DACA for L with time complexity (at most) 3.

Nevertheless, there is a 3-time one-sided 7/8-error PACA C' for L. Checking that the input $x = 0^k 1^l 2^m 3^n$ is such that (x is of the form $0^* 1^* 2^* 3^*$ and) $l, m \geq 2$ can be done without need of randomness simply by looking at the infixes of length 5 of x: Every cell collects the infix m that corresponds to its position in the input and rejects if m is disallowed. (We refer to [9, 14, 18] for the general method.) This procedure is carried out in parallel to the one we describe next (and a cell turns accepting if and only if it both procedures dictate it to do so).

Now we use randomness to check that one of $m \geq 3$ and $l \geq 3$ holds. In time step 1, every cell exposes its coin toss of step 0 so that its neighbors can read it and use it to choose their state in step 2. Let c_σ denote the leftmost cell in which $\sigma \in \Sigma$ appears, and let l_σ and r_σ be the coin tosses of the left and right neighbors of c_σ, respectively. We have c_1 accept if and only if $r_1 = 1$, c_3 if and only if $l_3 = 1$, and c_2 if and only if $l_2 + r_2 < 2$. All other cells accept regardless of the coin tosses they see (as long as x satisfies the conditions we specified above).

For $i \in \{1, 2, 3\}$, let A_i denote the event of cell c_i accepting. The above results in the following behavior: If $l = m = 2$, we have $r_1 = l_2$ and $r_2 = l_3$ since the coin tosses belong to the same cells, in which case C' never accepts. If $l \geq 3$ and r_2 and l_3 belong to the same cell (i.e., $r_2 = l_3$), then r_1 and l_2 do not belong to the same cell and we have

$$\Pr[C(x, U_{T\times|x|}) = 1] = \Pr[A_1] \Pr[A_2 \wedge A_3] = \Pr[r_1 = 1] \Pr[l_2 = 0 \wedge r_2 = l_3 = 1] = \frac{1}{8}.$$

The case $m \geq 3$ and r_1 and l_2 belonging to the same cell is similar. Finally, if $l \geq 3$ and $m \geq 3$, the values r_1, l_2, r_2, and l_3 are all independent and we have

$$\Pr[C(x, U_{T\times|x|}) = 1] = \prod_{i=1}^{3} \Pr[A_i] = \Pr[r_1 = 1] \Pr[l_2 + r_2 < 2] \Pr[l_3 = 1] > \frac{1}{8}.$$

B Proof of Theorem 6

This appendix contains the proof of Theorem 6, which we recall for the reader's convenience.

▶ **Theorem 6.** *The class of languages that can be accepted by a constant-time two-sided error PACA contains* $\mathsf{LLT}_{\cup\cap}$ *and is strictly contained in* LTT.

For the proof we first need to make the linear condition of LLT a bit more manageable:

▶ **Lemma 22.** *For any* $L = \mathsf{LLin}_\ell(\pi, \sigma, \alpha, \theta)$, *there is* α' *such that* $L = \mathsf{LLin}_\ell(\pi, \sigma, \alpha', \theta')$ *and:*

1. *The threshold* θ' *is equal to* 1.
2. *There is a constant* $\varepsilon > 0$ *such that, for every* $w \in \Sigma^*$, *we have either* $f'(w) < 1 - \varepsilon$ *or* $f'(w) > 1 + \varepsilon$, *where* $f'(w) = \sum_{m \in \Sigma^\ell} \alpha'(m) \cdot |w|_m$.
3. *There is* $k \in \mathbb{N}_0$ *such that, for every* m, *there is* $n \in [2^k]$ *such that* $\alpha'(m) = k - \log(n+1)$.

The first two items ensure the sum $f'(w)$ is always "far away" from 1. In turn, the third item enables us to represent $2^{-\alpha'(m)}$ using at most k (and, in particular, $O(1)$ many) bits.

Proof. The case where $\alpha(m) = 0$ for every m is trivial, so suppose there is some m for which $\alpha(m) > 0$. Because $|w|_m \in \mathbb{N}_0$ for every w and every m, $f(w) = \sum_{m \in \Sigma^\ell} \alpha(m) \cdot |w|_m$ is such that, given any $b \in \mathbb{R}_0^+$, there are only finitely many values of $f(w) \leq b$ (i.e., the set $\{f(w) \mid f(w) \leq b\}$ is finite). Hence, by setting

$$r = \frac{1}{2}\left(\max_{f(w) \leq \theta} f(w) + \min_{f(w) > \theta} f(w)\right)$$

and $\alpha''(m) = \alpha(m)/r$, we have that $f''(w) = \sum_{m \in \Sigma^\ell} \alpha''(m) \cdot |w|_m$ satisfies the second condition from the claim (i.e., for every $w \in \Sigma^*$, either $f''(w) < 1 - \varepsilon$ or $f''(w) > 1 + \varepsilon$) for some adequate choice of $\varepsilon > 0$.

Now we show how to satisfy the last condition without violating the first two. Essentially, we will choose k to be sufficiently large and then "round up" each $\alpha''(m)$ to the nearest value of the form $k - \log(n+1)$. To that end, let a be the minimal $\alpha''(m)$ for which $\alpha''(m) > 0$. In addition, let $k \in \mathbb{N}_0$ be such that $\alpha''(m) < k/2$ for every m and that

$$\log(2^{k/2} + 1) - \log(2^{k/2}) = \log(2^{k/2} + 1) - \frac{k}{2} \leq \frac{a\varepsilon}{2|\Sigma|^\ell}.$$

(This is possible because $\log(n+1) - \log n$ tends to zero as $n \to \infty$ and the right-hand side is constant.) Then, for every m, we set $\alpha'(m) = k - \log(n+1)$ where $n \in [2^k]$ is maximal such that $\alpha'(m) \geq \alpha''(m)$. By construction, $f'(w) \geq f''(w)$, so we need only argue that there is $\varepsilon' > 0$ such that, for every w for which $f''(w) < 1 - \varepsilon$, we also have $f'(w) < 1 - \varepsilon'$. In particular every said w must be such that, for every m, $|w|_m \leq 1/a$ (otherwise we would have $f''(w) > 1$). Noting that $|\alpha'(m) - \alpha''(m)|$ is maximal when $\alpha'(m) = k/2$ and $\alpha''(m) = k - \log(2^{k/2} + 1) + \delta$ for very small $\delta > 0$, we observe that

$$f'(w) = \sum_{m \in \Sigma^\ell} \alpha'(m) \cdot |w|_m < f''(w) + \frac{|\Sigma|^\ell}{a}\left(\log(2^{k/2} + 1) - \frac{k}{2}\right) < 1 - \varepsilon + \frac{\varepsilon}{2} = 1 - \frac{\varepsilon}{2},$$

that is, $f'(w) < 1 - \varepsilon'$ for $\varepsilon' = \varepsilon/2$, as desired. ◀

Proof of Theorem 6. We prove the two inclusions from the theorem's statement. The first one we address is that of $\mathsf{LLT}_{\cup\cap}$ in the class of constant-time two-sided error PACA.

First inclusion. Given $L = \mathsf{LLin}_\ell(\pi, \sigma, \alpha, \theta)$, we construct a constant-time two-sided error PACA C with $L(C) = L$. This suffices since by the closure properties shown in Proposition 19. We apply Lemma 22 and assume $\theta = 1$ and that there are k and ε as in the statement of Lemma 22. For simplicity, we also assume $\ell = 2k + 1$.

The automaton C operates in k steps as follows: Every cell sends its input symbol in both directions as a signal and, at the same time, aggregates the symbols it sees, thus allowing it to determine the initial configuration $m \in \Sigma^\ell$ of its k-neighborhood. Meanwhile, every cell also collects k random bits $r \in \{0,1\}^k$. The decision to accept is then simultaneously made in the k-th step, where a cell with k-neighborhood m accepts with probability $2^{-\alpha(m)}$ (independently of other cells). (This can be realized, for instance, by seeing r as the representation of a k-bit integer in $[2^k]$ and accepting if and only if $r \leq n$, where n is such that $2^{-\alpha(m)} = (n+1)/2^k$.) In the case of the first (resp., last) cell of C, it also checks that the prefix (resp., suffix) of the input is in π (resp., σ), rejecting unconditionally if this is not the case.

Hence, for an input word $w \in \Sigma^+$, the probability that C accepts is

$$\prod_{m \in \Sigma^\ell} \left(\frac{1}{2^{\alpha(m)}}\right)^{|w|_m} = 2^{-f(w)},$$

where $f(w) = \sum_{m \in \Sigma^\ell} \alpha(m) \cdot |w|_m$. Thus, if $w \in L$, then C accepts with probability $2^{-f(w)} > (1/2)^{1-\varepsilon}$; conversely, if $w \notin L$, the probability that C accepts is $2^{-f(w)} < (1/2)^{1+\varepsilon}$. Since ε is constant, we may apply Proposition 13 and reduce the error to $1/3$.

Second inclusion. The proof of the second inclusion is more involved. Let C be a T-time two-sided error PACA for some $T \in \mathbb{N}_+$. We shall obtain $L(C) \in \mathsf{LTT}$ in three steps:

1. The first step is a warm-up where the cells of C accept all independently from one another and that, if C accepts, then it does so in a fixed time step $t < T$.
2. Next we relax the requirement on independence between the cells by considering groups of cells (of maximal size) that may be correlated with one another regarding their acceptance.
3. Finally, we generalize what we have shown so that it also holds in the case where C may accept in any step $t < T$. This is the only part in the proof where closure under complement is required. (Here we use item 3 of Proposition 18.)

Step 1. Suppose that C only accepts in a fixed time step $t < T$ and that the events of any two cells accepting are independent from one another. We show that $L(C) = \mathsf{LLin}(\pi, \sigma, \alpha, \theta)$ for an adequate choice of parameters. Set $\ell = 2t + 1$ and let p_m be the probability that a cell with t-neighborhood $m \in \Sigma^\ell$ accepts in step t. In addition, let $\pi = \{p_{\ell-1}(w) \mid w \in L(C)\}$ and $\sigma = \{s_{\ell-1}(w) \mid w \in L(C)\}$ as well as $\theta = \log(3/2)$ and $\alpha(m) = \log(1/p_m)$ for $m \in \Sigma^\ell$. The probability that C accepts a word $w \in \Sigma^+$ is

$$\prod_{m \in \Sigma^\ell} p_m^{|w|_m} = \prod_{m \in \Sigma^\ell} \left(\frac{1}{2^{\alpha(m)}}\right)^{|w|_m} = 2^{-f(w)},$$

which is at least $2/3$ if and only if $f(w) \leq \theta$. It follows that $L(C) = \mathsf{LLin}_\ell(\pi, \sigma, \alpha, \theta)$.

Step 2. We now relax the requirements from the previous step so that the events of any two cells accepting need no longer be independent from one another. (C still only accepts in the fixed time step t.) Let $K = 2^{O(T^2)}$ be the upper bound from Lemma 15 and $\ell = 2(K+2)T$. Again, we set $\pi = \{p_{\ell-1}(w) \mid w \in L(C)\}$, $\sigma = \{s_{\ell-1}(w) \mid w \in L(C)\}$, and $\theta = \log(3/2)$. As for $\alpha(m)$, we set $\alpha(m) = 0$ unless m is such that there is $d \leq K$ with

$$m = abr_1 s_1 r_2 s_2 \cdots r_d s_d c$$

where $a \in \Sigma^T$ is arbitrary, $b \in \Sigma^{2T}$ contains *no critical cells*, each $r_j \in \Sigma$ is a critical cell (for step t), the $s_j \in \Sigma^{\leq 2T-1}$ are arbitrary, and $c \in \Sigma^*$ has length $|c| \geq T$ and, if c contains any critical cell, then this cell accepts independently from r_d. In addition, we require c to be of maximal length with this property.

Note we need a as context to ensure that b indeed does not contain critical cells (since determining this requires knowledge of the states in the T-neighborhood of the respective cell); the same holds for r_d and c. By construction and Lemma 11, the group of cells $r_1, \dots, r_d$ is such that (although its cells are not necessarily independent from one another) its cells accepts independently from any other critical cell in C. Furthermore, b ensures m aligns properly with the group and that the group does not appear in any other infix. Letting p_m be the probability that every one of the r_j accept, for m as above we set $\alpha(m) = \log(1/p_m)$. Then, as before, the probability that C accepts a word $w \in \Sigma^+$ is $2^{-f(w)}$, which is at least $2/3$ if and only if $f(w) \leq \theta$, thus implying $L(C) = \mathsf{LLin}_\ell(\pi, \sigma, \alpha, \theta)$.

Step 3. In this final step we generalize the argument so it also applies to the case where C may accept in any time step $t < T$. First note that, given any $p > 0$, if we set $\theta = \log(1/p)$ in the second step above (instead of $\log(3/2)$), then we have actually shown that

$$\{w \in \Sigma^+ \mid \Pr[C \text{ accepts } w \text{ in step } t] \geq p\} = \mathsf{LLin}_\ell(\pi, \sigma, \alpha, \theta).$$

In fact, we can generalize this even further: Given any $\varnothing \neq \tau \subseteq [T]$, by setting α adequately we can consider the acceptance probability for the steps in τ altogether:[8]

$$L(\tau, p) = \{w \in \Sigma^+ \mid \Pr[C \text{ accepts } w \text{ in every step } t \in \tau] \geq p\} = \mathsf{LLin}_\ell(\pi, \sigma, \alpha, \theta).$$

This is because the bound on critical cells of Lemma 15 holds for all steps where C accepts with non-zero probability and, in addition, as defined above m already gives enough context to check if the respective critical cells also accept in any previous step. (That is, we construct m as above by using $t = \max \tau$; however, since the sets of critical cells for different time steps may not be identical, we must also relax the condition for the r_i so that r_i need only be a critical cell in at least one of the time steps of τ.) Since LTT is closed under complement, we then also have

$$\overline{L(\tau, p)} = \{w \in \Sigma^+ \mid \Pr[C \text{ accepts } w \text{ in every step } t \in \tau] < p\} \in \mathsf{LTT}.$$

Fix some input word $w \in \Sigma^+$ to C. For $\varnothing \neq \tau \subseteq [T]$, let Z_τ denote the event where C accepts w in every step $t \in \tau$. By the inclusion-exclusion principle, we have

$$\Pr[C \text{ accepts } w] = \Pr\left[\exists t \in [T] : Z_{\{t\}}\right] = \sum_{\substack{\tau \subseteq [T] \\ |\tau|=1}} \Pr[Z_\tau] - \sum_{\substack{\tau \subseteq [T] \\ |\tau|=2}} \Pr[Z_\tau] + \cdots + (-1)^{T+1} \Pr[Z_{[T]}]$$

(where the probabilities are taken over the coin tosses of C). This means that, if we are somehow given values for $p(\tau) = \Pr[Z_\tau]$ so that the sum above is at least $2/3$, then we can intersect a finite number of $L(\tau, p(\tau))$ languages and their complements and obtain some language that is guaranteed to contain only words in $L(C)$. Concretely, let $p(\tau) \geq 0$ for every $\varnothing \neq \tau \subseteq [T]$ be given so that

$$\sum_{\substack{\varnothing \neq \tau \subseteq [T] \\ |\tau| \text{ odd}}} p(\tau) - \sum_{\substack{\varnothing \neq \tau \subseteq [T] \\ |\tau| \text{ even}}} p(\tau) \geq \frac{2}{3}.$$

[8] Of course we are being a bit sloppy here since Definition 9 demands that a PACA should halt whenever it accepts. What is actually meant is that, having fixed some random input, if we extend the space-time diagram of C on input w so that it spans all of its first T steps (simply by applying the transition function of C), then, for every $t \in \tau$, the t-th line in the diagram contains only accepting cells.

Let $\mathcal{T}_{\text{odd}} = \{\tau \subseteq [T] \mid \tau \neq \varnothing, |\tau| \text{ odd}, p(\tau) > 0\}$ and $\mathcal{T}_{\text{even}} = \{\tau \subseteq [T] \mid \tau \neq \varnothing, |\tau| \text{ even}, p(\tau) > 0\}$. Then necessarily

$$L(p) = \left(\bigcap_{\tau \in \mathcal{T}_{\text{odd}}} L(\tau, p(\tau)) \right) \cap \left(\bigcap_{\tau \in \mathcal{T}_{\text{even}}} \overline{L(\tau, p(\tau))} \right) \subseteq L(C)$$

contains every $w \in L(C)$ for which $\Pr[Z_\tau] \geq p(\tau)$ for $\tau \in \mathcal{T}_{\text{odd}}$ and $\Pr[Z_\tau] \leq p(\tau)$ for $\tau \in \mathcal{T}_{\text{even}}$. The key observation is that *there are only finitely many values the* $\Pr[Z_\tau]$ *may assume*. This is because Z_τ only depends on a finite number of coin tosses, namely the ones in the lightcones of the cells that are critical in at least one of the steps in τ (which, again, is finite due to Lemma 15). Hence, letting P denote the set of all possible mappings of the τ subsets to these values, we may write

$$L(C) = \bigcup_{p \in P} L(p) \in \mathsf{LTT}.$$

Strictness of inclusion. The final statement left to prove is that the inclusion just proven is proper. This is comparatively much simpler to prove. We show that the language

$$L = \{w \in \{0,1\}^+ \mid |w|_1 \geq 2\} \in \mathsf{LTT}$$

cannot be accepted by two-sided error PACA in constant time. For the sake of argument, assume there is such a PACA C with time complexity $T \in \mathbb{N}_+$. Consider which cells in C are critical based on their initial local configuration. Certainly a cell with an all-zeroes configuration 0^{2T-1} cannot be critical. Since $0^n 10^n \notin L(C)$ for any n (but $0^n 10^n 1 \in L(C)$), there must be m_1, m_2 so that $m_1 + m_2 = 2T - 2$ and $c = 0^{m_1} 10^{m_2}$ is the initial local configuration of a critical cell. This means that in

$$x = 0^{2T}(c0^{2T})^{T2^T} \in L$$

we have at least 2^T cells in x that are critical for the same time step $t \in [T]$ (by an averaging argument) and that are also all independent from one another (by Lemma 11). In turn, this implies the following, which contradicts $x \in L(C)$:

$$\Pr[C \text{ accepts } x] \leq \left(1 - 2^{-T}\right)^{2^T} < \frac{1}{e} < \frac{2}{3}. \qquad \blacktriangleleft$$