Unambiguous Constrained Automata

Michaël Cadilhac[1], Alain Finkel[2,⋆], and Pierre McKenzie[1,⋆⋆]

[1] DIRO, Université de Montréal
C.P. 6128 succ. Centre-Ville, Montréal (Québec), H3C 3J7 Canada
{cadilhac,mckenzie}@iro.umontreal.ca
[2] LSV, ENS Cachan, CNRS
61 avenue du Président Wilson, 94235 Cachan Cedex, France
finkel@lsv.ens-cachan.fr

Abstract. The class of languages captured by Constrained Automata (CA) that are unambiguous is shown to possess more closure properties than the provably weaker class captured by deterministic CA. Problems decidable for deterministic CA are nonetheless shown to remain decidable for unambiguous CA, and testing for *regularity* is added to this set of decidable problems. Unambiguous CA are then shown incomparable with deterministic reversal-bounded machines in terms of expressivity, and a *deterministic* model equivalent to unambiguous CA is identified.

Keywords: unambiguity, constrained automata, regularity test.

1 Introduction

A recent trend in automata theory is to study flavors of nondeterminism, which are introduced to provide a scale of expressiveness in different models (see [4] for a survey). The usual goal is to strike a balance between the expressiveness of nondeterministic models and the undecidability properties that often come with nondeterminism. A natural restriction to nondeterminism is *unambiguity*, i.e., the property that despite the underlying nondeterminism, there be at most one way to accept an input word. Within the context of finite automata, unambiguity and nondeterminism are equally expressive, but many open problems concerning the state complexity of unambiguity remain. Within more general contexts, the first question is often whether unambiguity offers more expressiveness than determinism; if so, then the examination of the closure and decidability properties of the new class often reveals that it inherits good properties. Another line of attack is to find a deterministic model equivalent to an unambiguous model, so as to understand how unambiguity affects a given model.

In [9], Klaedtke and Rueß studied Constrained Automata (CA),[1] a model whose expressive power lies between regular languages and context-sensitive

⋆ Ce travail a bénéficié d'une aide de l'Agence Nationale de la Recherche portant la référence "REACHARD-ANR-11-BS02-001".
⋆⋆ Supported by the Natural Sciences and Engineering Research Council of Canada.
[1] In [9], the model under study is called *Parikh automata*. CA are but an effectively equivalent model with an arguably simpler definition.

H.-C. Yen and O.H. Ibarra (Eds.): DLT 2012, LNCS 7410, pp. 239–250, 2012.
© Springer-Verlag Berlin Heidelberg 2012

languages [3]. Klaedtke and Rueß successfully used the CA in the model checking of hardware circuits, suggesting that CA is a model of interest for real-life applications. The deterministic variant (DetCA) of the CA enjoys more closure properties (e.g., complement) and decidability properties (e.g., universality) than the CA, but is unable to express languages as simple as $\{a, b\}^* \cdot \{a^n b^n \mid n \geq 1\}$ [3]. Buoyed by Colcombet's recent systematic examination of unambiguity [4], here we initiate the study of unambiguous CA (UnCA).

We show that UnCA enjoy more closure properties than DetCA, while being more expressive. The class of languages UnCA defines is indeed closed under Boolean operations, reversal, and right and left quotient. We show that the problems known to be decidable for DetCA (emptiness, universality, finiteness, inclusion) remain decidable for UnCA. As the main technical result of this paper, we show that regularity is decidable for UnCA; by contrast, regularity is known to be undecidable for CA [3], while its status was unknown for DetCA. Finally, although DetCA are less powerful than UnCA, we present a natural *deterministic* model equivalent to UnCA; as a result of independent interest, we show that the nondeterministic variant of this model has the same expressive power as CA.

Section 2 contains preliminaries, settles notation, and defines the models in play. Section 3 investigates the closure and expressiveness properties of UnCA and compares it to deterministic reversal-bounded counter machines. Section 4 proceeds with the decidability properties of UnCA and proves regularity decidable. Section 5 shows that there is a natural deterministic model equivalent to UnCA. Section 6 concludes with a brief discussion.

2 Preliminaries

Integers, Vectors, Monoids. We write $\mathbb{N}$ for the nonnegative integers. Let $d \geq 1$. Vectors in $\mathbb{N}^d$ are noted in bold, e.g., $\boldsymbol{v}$ whose elements are $v_1, v_2, \ldots, v_d$. We write $\boldsymbol{e_i} \in \{0, 1\}^d$ for the vector having a 1 only in position i and $\mathbf{0}$ for the all-zero vector. We view $\mathbb{N}^d$ as the additive monoid $(\mathbb{N}^d, +)$, with $+$ the component-wise addition and $\mathbf{0}$ the identity element. Given an order on some set $\Sigma = \{a_1, a_2, \ldots, a_n\}$ we often refer to the components of a vector $\boldsymbol{v} \in \mathbb{N}^{|\Sigma|}$ by x_{a_i} instead of x_i. In particular, for $a \in \Sigma$, x_a refers to the i-th component of $\boldsymbol{x}$ where i is such that $a_i = a$. Let $s \geq 0$ and $p \geq 1$, we define the congruence $\equiv_{s,p}$, by $x \equiv_{s,p} y$ iff $(x = y < s) \vee (x, y \geq s \wedge x = y \pmod p)$, for $x, y \in \mathbb{N}$; we write $[x]_{s,p}$ for the equivalence class of x under $\equiv_{s,p}$. We extend $\equiv_{s,p}$ component-wise to vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{N}^d$ by letting $\boldsymbol{x} \equiv_{s,p} \boldsymbol{y}$ iff $x_i \equiv_{s,p} y_i$ for all $1 \leq i \leq d$; similarly, $[\boldsymbol{x}]_{s,p}$ is the equivalence class of $\boldsymbol{x}$ under this relation.

For a monoid $(M, \cdot)$ and $S \subseteq M$, we write S^* for the monoid generated by S, i.e., the smallest submonoid of $(M, \cdot)$ containing S. A (monoid) *morphism* from $(M, \cdot)$ to $(N, \circ)$ is a function $h \colon M \to N$ such that $h(m_1 \cdot m_2) = h(m_1) \circ h(m_2)$, and, with e_M (resp. e_N) the identity element of M (resp. N), $h(e_M) = e_N$. Moreover, if $M = S^*$ for some finite set of symbols S (and this will always be the case), then h need only be defined on the elements of S. In this case, h is said to be *erasing* if there is an $s \in S$ such that $h(s) = e_N$. If in addition $N = T^*$

for some finite set of symbols T, h is said to be *length-preserving* if for all $s \in S$, $h(s) \in T$.

Semilinear Sets, Parikh Image. A subset C of $\mathbb{N}^d$ is *linear* if there exist $\boldsymbol{c} \in \mathbb{N}^d$ and a finite $P \subseteq \mathbb{N}^d$ such that $C = \boldsymbol{c} + P^*$. The subset C is said to be *semilinear* if it is equal to a finite union of linear sets: $\{4n + 56 \mid n > 0\}$ is semilinear while $\{2^n \mid n > 0\}$ is not. We will often use the fact that the semilinear sets are those sets of natural numbers definable in first-order logic with addition [5]. A semilinear set is said to be *effectively semilinear* if its description as a set of $\boldsymbol{c}$'s and P's, or equivalently as a formula, can be computed from the data at hand. Let $\Sigma = \{a_1, a_2, \ldots, a_n\}$ be an (ordered) alphabet with ε the empty word. The *Parikh image* is the morphism $\Phi\colon \Sigma^* \to \mathbb{N}^n$ defined by $\Phi(a_i) = \boldsymbol{e_i}$, for $1 \leq i \leq n$ — in particular, we have that $\Phi(\varepsilon) = \mathbf{0}$. For $w \in \Sigma^*$, with $\Phi(w) = \boldsymbol{x}$ and $a \in \Sigma$, we write $|w|_a$ for x_a. The Parikh image of a language L is defined as $\Phi(L) = \{\Phi(w) \mid w \in L\}$. The name of this morphism stems from Parikh's theorem [11], stating that for L context-free, $\Phi(L)$ is semilinear. For $L \subseteq \Sigma^*$ and $C \subseteq \mathbb{N}^n$, define $L\restriction_C = \{w \in L \mid \Phi(w) \in C\}$.

Languages, Operations. For $u = u_1 u_2 \cdots u_n \in \Sigma^*$, define $u^{\mathrm{R}} = u_n \cdots u_2 u_1$ as the *reversal* of u. For $L_1, L_2 \subseteq \Sigma^*$, define L_1^{R} as the set of the reversals of each word in L_1; $(L_1)^{-1}L_2 = \{v \mid (\exists u \in L_1)[u \cdot v \in L_2]\}$ as the *left quotient* of L_2 by L_1; and $L_1(L_2)^{-1} = \{u \mid (\exists v \in L_2)[u \cdot v \in L_1]\}$ as the *right quotient* of L_1 by L_2. A language $L \subseteq \Sigma^*$ is *bounded* [6] if there exist $n > 0$ and a sequence of words $w_1, w_2, \ldots, w_n \in \Sigma^+$, which we call *a socle of* L, such that $L \subseteq w_1^* w_2^* \cdots w_n^*$. The *iteration set* of L w.r.t. this socle is (uniquely) defined as $\mathsf{Iter}_{(w_1,w_2,\ldots,w_n)}(L) = \{(i_1, i_2, \ldots, i_n) \in \mathbb{N}^n \mid w_1^{i_1} w_2^{i_2} \cdots w_n^{i_n} \in L\}$.

Automata. An automaton is a quintuple $A = (Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, Σ is an alphabet, $\delta \subseteq Q \times \Sigma \times Q$ is a set of transitions, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is a set of final states. For a transition $t = (q, a, q') \in \delta$, we write $t = q\text{-}a\text{-}q'$ and define $\mathsf{From}(t) = q$ and $\mathsf{To}(t) = q'$. We define $\mu_A\colon \delta^* \to \Sigma^*$ as the length-preserving morphism given by $\mu_A(t) = a$, with, in particular, $\mu_A(\varepsilon) = \varepsilon$, and write μ when A is clear from the context. A *path* π on A is a word $\pi = t_1 t_2 \cdots t_n \in \delta^*$ such that $\mathsf{To}(t_i) = \mathsf{From}(t_{i+1})$ for $1 \leq i < n$; we extend From and To to paths, letting $\mathsf{From}(\pi) = \mathsf{From}(t_1)$ and $\mathsf{To}(\pi) = \mathsf{To}(t_n)$. We say that $\mu(\pi)$ is the *label* of π. A path π is *initial* if $\mathsf{From}(\pi) = q_0$, *final* if $\mathsf{To}(\pi) \in F$, and *accepting* if it is both initial and final; we write $\mathsf{Run}(A)$ for the language over δ of accepting paths (or *runs*) on A. We write $L(A)$ for the language of A, i.e., the labels of the accepting paths. The automaton A is *deterministic* if $(p\text{-}a\text{-}q \in \delta \wedge p\text{-}a\text{-}q' \in \delta)$ implies $q = q'$. An ε-automaton is an automaton $A = (Q, \Sigma, \delta, q_0, F)$ as above, except with $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q$ so that in particular μ_A becomes an erasing morphism. An (ε-)automaton A is *unambiguous* if each word in $L(A)$ is the label of only one path in $\mathsf{Run}(A)$.

Affine Functions. A function $f\colon \mathbb{N}^d \to \mathbb{N}^d$ is a (total and positive) *affine function* of dimension d if there exist a matrix $M \in \mathbb{N}^{d \times d}$ and $\boldsymbol{v} \in \mathbb{N}^d$ such that for any $\boldsymbol{x} \in \mathbb{N}^d$, $f(\boldsymbol{x}) = M.\boldsymbol{x} + \boldsymbol{v}$. We abusively write $f = (M, \boldsymbol{v})$. We let $\mathcal{F}_d$ be the monoid of such functions under the operation $\diamond$ defined by $(f \diamond g)(\boldsymbol{x})$

$= g(f(\boldsymbol{x}))$, where the identity element is the identity function, i.e., $(Id, \mathbf{0})$ with Id the identity matrix of dimension d. Let U be a monoid morphism from Σ^* to $\mathcal{F}_d$. For $w \in \Sigma^*$, we write U_w for $U(w)$, so that the application of $U(w)$ to a vector $\boldsymbol{v}$ is written $U_w(\boldsymbol{v})$, and U_ε is the identity function. We define $\mathcal{M}(U)$ as the multiplicative matrix monoid generated by the matrices used to define U, i.e., $\mathcal{M}(U) = \{M \mid (\exists a \in \Sigma)(\exists \boldsymbol{v})[U_a = (M, \boldsymbol{v})]\}^*$.

Definition 1 (Constrained automaton [3]). *A* constrained automaton (CA) *is a pair (A, C) where A is an ε-automaton with d transitions and $C \subseteq \mathbb{N}^d$ is semilinear. Its language is $L(A, C) = \mu_A(\mathsf{Run}(A)\restriction_C)$. The CA is said to be:*
- Deterministic (DetCA) *if A is a deterministic automaton;*
- Unambiguous (UnCA) *if A is an unambiguous ε-automaton.*

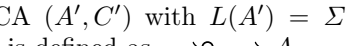
We write $\mathcal{L}_{\mathrm{CA}}$, $\mathcal{L}_{\mathrm{DetCA}}$,[2] and $\mathcal{L}_{\mathrm{UnCA}}$ for the classes of languages recognized by CA, DetCA, and UnCA, respectively.

3 Closure Properties and Expressiveness of UnCA

We show closure and nonclosure properties, and we give languages witnessing the strictness of $\mathcal{L}_{\mathrm{DetCA}} \subsetneq \mathcal{L}_{\mathrm{UnCA}} \subsetneq \mathcal{L}_{\mathrm{CA}}$. Lemma 1 is a tool that will prove useful when combining UnCA. It is shown by applying the standard procedure of backward-closure (e.g., [12]) and keeping track of the closure in the constraint set:

Lemma 1. *For any UnCA (A, C), there is an UnCA (A', C') where A' has no ε-transition, $L(A) = L(A')$, and $L(A, C) = L(A', C')$.*

Proposition 1. *$\mathcal{L}_{\mathrm{UnCA}}$ is closed under union.*

Proof (sketch). First, we note that for an UnCA (A, C) over the alphabet Σ, there is an UnCA (A', C') with $L(A') = \Sigma^*$ and $L(A', C') = L(A, C)$. The ε-automaton A' is defined as  where $\overline{A}$ is a deterministic automaton for $\overline{L(A)}$ and the two new transitions are labeled by ε. Then C' is defined to reject if the transition to $\overline{A}$ is taken, and to accept if the run is in A and its Parikh image is in C. Clearly, A' is unambiguous.

Now let (A, C) and (B, D) be two UnCA over the same alphabet Σ (w.l.o.g.), and with $L(A) = L(B) = \Sigma^*$, as per the previous discussion. We design an automaton that runs A and B in parallel. We rely on Lemma 1 to synchronize the two automata. For any word w, there will be exactly one way to read w over A and B, thus only one way to read w over both at the same time. Finally, we constrain this automaton by extracting the paths in A and B and checking that at least one of them is in its respective constraint set. □

As $\overline{L(A, C)} = \overline{L(A)} \cup L(A, \overline{C})$, we have:

Proposition 2. *$\mathcal{L}_{\mathrm{UnCA}}$ is closed under complement and intersection.*

Note that $\mathcal{L}_{\mathrm{DetCA}}$ is not closed under reversal, as $\{a, b\}^* \cdot \{a^n b^n \mid n \geq 1\}$ is not in $\mathcal{L}_{\mathrm{DetCA}}$ while its reversal is [3]. Thus it is a curiosity, especially for a class described by a deterministic model (forthcoming Theorem 4), that we have:

[2] In [3], $\mathcal{L}_{\mathrm{CA}}$ and $\mathcal{L}_{\mathrm{DetCA}}$ are written $\mathcal{L}_{\mathrm{PA}}$ and $\mathcal{L}_{\mathrm{DetPA}}$, in reference to Parikh automata [9], which are equivalent to CA.

Proposition 3. $\mathcal{L}_{\mathrm{UnCA}}$ *is closed under reversal.*

Proof. Let (A, C) be an UnCA. Let B be the ε-automaton A in which a fresh state q_{f} is set to be the only final state, and with a transition from each former final state to q_{f} labeled ε. Clearly, B is unambiguous. Adjust C into C' so that the added transitions in B do not affect the acceptance of a word, i.e., $L(B, C') = L(A, C)$. Then define D as the ε-automaton B in which every transition is reversed, i.e., (q, a, q') is a transition of B iff (q', a, q) is a transition of D; the order on the transition set of D is the same as that of B. Additionally, set q_{f} as the initial state and the former initial state of B as the only final state. Then D is unambiguous: clearly, $\mathsf{Run}(B) = \{\pi^{\mathrm{R}} \mid \pi \in \mathsf{Run}(D)\}$, thus the accepting paths in D labeled w are the reversal of the accepting paths in B labeled w^{R}. As B is unambiguous, only one such path may exist, thus D is unambiguous. Hence $L(D, C') = (L(B, C'))^{\mathrm{R}} = (L(A, C))^{\mathrm{R}}$. □

Proposition 4. *Let* $L_1 \in \mathcal{L}_{\mathrm{CA}}$ *and* $L_2 \in \mathcal{L}_{\mathrm{UnCA}}$. *Then* $L_1^{-1} L_2 \in \mathcal{L}_{\mathrm{UnCA}}$.

Proof. Let (A, C) be a CA, (B, D) be an UnCA, with $A = (Q_A, \Sigma, \delta_A, q_{0,A}, F_A)$ and $B = (Q_B, \Sigma, \delta_B, q_{0,B}, F_B)$. We suppose, thanks to Lemma 1, that no transition of B is labeled by ε, and that each state of B is reachable from $q_{0,B}$ and can reach a final state. For $q \in Q_B$, define $B^{\to q}$ (resp. $B^{q\to}$) to be the ε-automaton B where the initial state (resp. the only final state) is q, and note that $B^{\to q}$ is unambiguous, as any path from q to a final state can be prefixed with a path from $q_{0,B}$ to q to make an accepting path in B. First note that:

Claim 1. For any $q_B \in Q_B$, the set $E^{q_B} = \{(\Phi(\pi), \Phi(\rho)) \mid \pi \in \mathsf{Run}(A) \wedge \rho \in \mathsf{Run}(B^{q_B\to}) \wedge \mu_A(\pi) = \mu_B(\rho)\}$ is effectively semilinear.

A word w is in $(L(A, C))^{-1} L(B, D)$ iff there is a state $q_B \in Q_B$ and a word $u \in L(A, C)$ such that $u \in L(B^{q_B\to})$, $w \in L(B^{\to q_B})$, and the Parikh image of one (in fact, the only) path for u in $B^{q_B\to}$ concatenated with the path for w in $B^{\to q_B}$ is in D. This is the case iff there is a state $q_B \in Q_B$ and a pair $(\boldsymbol{x}, \boldsymbol{y}) \in E^{q_B}$ such that $\boldsymbol{x} \in C$ and the Parikh image $\boldsymbol{z}$ of the only path in $B^{\to q_B}$ labeled w plus $\boldsymbol{y}$ is in D. In symbols, a word w is in $(L(A, C))^{-1} L(B, D)$ iff it is in $\bigcup_{q_B \in Q_B} L(B^{\to q_B},\ \{\boldsymbol{z} \mid (\exists (\boldsymbol{x}, \boldsymbol{y}) \in E^{q_B})[\boldsymbol{x} \in C \wedge \boldsymbol{y} + \boldsymbol{z} \in D]\})$. □

Remark 1. We note that a similar proof shows that $\mathcal{L}_{\mathrm{UnCA}}$ is closed under right quotient. Also, similar proofs show that $\mathcal{L}_{\mathrm{DetCA}}$ is closed under both right and left quotient, settling those two closure properties that were left open in [9].

Let $P_1 = \{w = w_1 w_2 \cdots w_k \in \{\sqsubset, \sqsupset\}^* \mid (\forall i)[|w_1 w_2 \cdots w_i|_{\sqsubset} \geq |w_1 w_2 \cdots w_i|_{\sqsupset}]\}$ be the prefixes of the semi-Dyck language with one set of parentheses. Then:

Proposition 5. $P_1 \notin \mathcal{L}_{\mathrm{CA}}$ *and* $\overline{P_1} \in \mathcal{L}_{\mathrm{CA}} \setminus \mathcal{L}_{\mathrm{UnCA}}$.

Proof. (Sketch: $P_1 \notin \mathcal{L}_{\mathrm{CA}}$ and $\overline{P_1} \in \mathcal{L}_{\mathrm{CA}}$.) An expressiveness lemma for CA similar to [3, Lemma 1] shows that $P_1 \notin \mathcal{L}_{\mathrm{CA}}$. Moreover, we can design a simple CA for $\overline{P_1}$ which guesses a position in the input word at which the number of $\sqsubset$'s read so far is less than the number of $\sqsupset$'s.

($\overline{P_1} \notin \mathcal{L}_{\mathrm{UnCA}}$.) If $\overline{P_1} \in \mathcal{L}_{\mathrm{UnCA}}$, then $P_1 \in \mathcal{L}_{\mathrm{UnCA}}$ (Proposition 2), but as $\mathcal{L}_{\mathrm{UnCA}} \subseteq \mathcal{L}_{\mathrm{CA}}$, this contradicts $P_1 \notin \mathcal{L}_{\mathrm{CA}}$. □

Theorem 1. $\mathcal{L}_{\mathrm{DetCA}} \subsetneq \mathcal{L}_{\mathrm{UnCA}} \subsetneq \mathcal{L}_{\mathrm{CA}}$.

Proof. The chain of inclusion is immediate. The strictness of $\mathcal{L}_{\mathrm{DetCA}} \subsetneq \mathcal{L}_{\mathrm{UnCA}}$ is witnessed by $\{a,b\}^* \cdot \{a^n b^n \mid n \geq 1\}$, as previously mentioned, and the strictness of $\mathcal{L}_{\mathrm{UnCA}} \subsetneq \mathcal{L}_{\mathrm{CA}}$ is witnessed by $\overline{P_1}$ (Proposition 5). □

Proposition 6. $\mathcal{L}_{\mathrm{UnCA}}$ *is neither closed under concatenation with a regular language, nor under length-preserving morphisms, nor under starring.*

Proof. (Concatenation.) Let $\Sigma = \{\sqsubset, \sqsupset\}$. The language $L_< = \{w \in \Sigma^* \mid |w|_\sqsubset < |w|_\sqsupset\}$ is in $\mathcal{L}_{\mathrm{DetCA}}$ and such that $\overline{P_1} = L_< \cdot \Sigma^* \notin \mathcal{L}_{\mathrm{UnCA}}$.

(Length-preserving morphisms and starring.) Let $T = \{\mathbb{L}, \mathbb{J}\}$, then $L_< \cdot T^* \in \mathcal{L}_{\mathrm{UnCA}}$. The length-preserving morphism $h\colon (\Sigma \cup T)^* \to \Sigma^*$ defined by $h(\mathbb{L}) = h(\sqsubset) = \sqsubset$, $h(\mathbb{J}) = h(\sqsupset) = \sqsupset$ is such that $h(L_< \cdot T^*) = L_< \cdot \Sigma^* \notin \mathcal{L}_{\mathrm{UnCA}}$. For starring, it is shown in [3, Proposition 10] that with $L = \{a^n b^n \mid n \in \mathbb{N}\} \in \mathcal{L}_{\mathrm{DetCA}}$, $L^* \notin \mathcal{L}_{\mathrm{CA}} \supsetneq \mathcal{L}_{\mathrm{UnCA}}$. □

UnCA and RBCM. It is known that *one-way reversal-bounded counter machines* (RBCM) [8] are as powerful as CA [9], while deterministic such machines (DetRBCM) are more powerful than DetCA [3].

Definition 2 (RBCM [8]). *A* one-way counter machine *is a finite-state read-only device that decides at each point whether to move its input head one step to the right and uses a finite number of* counters *holding natural numbers, which can be incremented, decremented, and tested for* 0. *It is* reversal-bounded *(RBCM) if there is a constant r such that each accepting run changes between increment and decrement at most r times for each counter. It is* deterministic *(DetRBCM) if at any point the next values of the counters and the device's state are uniquely determined by the symbol currently read, the counter values, and the device's state. We write $\mathcal{L}_{\mathrm{DetRBCM}}$ for the class of languages recognized by DetRBCM.*

Proposition 7. $\mathcal{L}_{\mathrm{DetRBCM}}$ *and* $\mathcal{L}_{\mathrm{UnCA}}$ *are incomparable.*

Proof (sketch). A DetRBCM can deterministically use extra information provided in the input word to check for a certain property later in the input; this is illustrated by $L = \{a^n w \mid w \in \{\sqsubset, \sqsupset\}^* \wedge |w_1 w_2 \cdots w_n|_\sqsubset < |w_1 w_2 \cdots w_n|_\sqsupset\} \in \mathcal{L}_{\mathrm{DetRBCM}}$.

Suppose $L \in \mathcal{L}_{\mathrm{UnCA}}$. Proposition 4 then asserts that $(\{a\}^*)^{-1} L \cap \{\sqsubset, \sqsupset\}^*$ is in $\mathcal{L}_{\mathrm{UnCA}}$. But this latter language is $\overline{P_1} \notin \mathcal{L}_{\mathrm{UnCA}}$ (Proposition 5), a contradiction.

In the other direction, $\{a,b\}^* \cdot \{a^n b^n \mid n \geq 1\} \in \mathcal{L}_{\mathrm{UnCA}} \setminus \mathcal{L}_{\mathrm{DetRBCM}}$ [3,2]. □

4 Decision Problems for UnCA

We recall the following decidability results, that hold equally well for UnCA:

Proposition 8 ([9,3]). *Given a CA, it is decidable whether its language is empty, and whether its language is finite.*

With the closure properties of $\mathcal{L}_{\mathrm{UnCA}}$ of Proposition 2, this implies:

Proposition 9. *Given an UnCA, it is decidable whether its language is Σ^*. Given two UnCA, it is decidable whether the language of the first is included in the language of the second.*

The rest of this section is devoted to the main technical result of our paper, namely that it is decidable whether the language of an UnCA is regular. Our technique is mainly in two steps: we first show that it is decidable whether a *bounded* CA language (given additionally a socle of the language) is regular (Lemma 3) then reduce the decision in the general case to the decision with bounded CA languages.

Definition 3 ([7]). *A set C is* unary *if it is equal to a finite union of linear sets, each period of each linear set having at most one nonzero coordinate.*

Lemma 2 ([7, Theorem 1.3]). *Let $L \subseteq w_1^* w_2^* \cdots w_n^*$. The language L is regular iff $\mathsf{Iter}_{(w_1,w_2,\ldots,w_n)}(L)$ is unary.*

Lemma 3. *Given a CA (A, C) and words $w_1, w_2, \ldots, w_n$ such that $L(A, C) \subseteq w_1^* w_2^* \cdots w_n^*$, it is decidable whether $L(A, C)$ is regular.*

Proof. Let (A, C) be a CA with $L(A, C) \subseteq w_1^* w_2^* \cdots w_n^*$. Let $T = \{a_1, a_2, \ldots, a_n\}$ be a set of fresh symbols and define the morphism $h\colon T^* \to \Sigma^*$ by $h(a_i) = w_i$ for all i. Now let (A', C') be the CA with language $h^{-1}(L(A, C)) \cap a_1^* a_2^* \cdots a_n^*$ obtained by the (effective) closures of CA. Then for $\boldsymbol{i} \in \mathbb{N}^n$, $a_1^{i_1} a_2^{i_2} \cdots a_n^{i_n} \in L(A', C')$ iff $w_1^{i_1} w_2^{i_2} \cdots w_n^{i_n} \in L(A, C)$; hence the Parikh image of $L(A', C')$ is $\mathsf{Iter}_{(w_1,w_2,\ldots,w_n)}(L(A, C))$. Now $\Phi(L(A', C'))$ is an effectively semilinear set [9, Lemma 5], hence we can decide whether it is a unary set (see [7, Section 3]). This amounts to deciding, by Lemma 2, whether $L(A, C)$ is regular. □

Lemma 4. *The language of an UnCA (A, C) is regular iff $\mathsf{Run}(A)\restriction_C$ is regular.*

Proof. First, suppose $\mathsf{Run}(A) \restriction_C$ is regular, for a CA (A, C). As $L(A, C) = \mu(\mathsf{Run}(A) \restriction_C)$ and regular languages are closed under morphisms, $L(A, C)$ is regular. This part does not rely on unambiguity.

Second, consider an UnCA (A, C). We remark that if an accepting path of A is labeled by a word in $L(A, C)$, then it is in $\mathsf{Run}(A)\restriction_C$ (the converse is true of any CA). Indeed, since a path labeled by a word w in $L(A, C)$ is, by unambiguity, the only path labeled w in $\mathsf{Run}(A)$, it has its Parikh image in C. In other words, $\mathsf{Run}(A) \restriction_C = \mu^{-1}(L(A, C)) \cap \mathsf{Run}(A)$. Now, as the class of regular languages is closed under inverse morphisms and intersection, if $L(A, C)$ is regular then $\mathsf{Run}(A)\restriction_C$ is regular. □

Remark 2. The inclusion $\mathsf{Run}(A)\restriction_C \supseteq \mu^{-1}(L(A, C)) \cap \mathsf{Run}(A)$ is crucial to the proof of Lemma 4 and to the decidability of regularity for UnCA. Indeed, both this inclusion and Lemma 4 fail for CA — in fact, regularity is undecidable for CA [3]. For example, let A be the automaton: with r initial. Define C to constrain the two loops on r and s to occur the same number of times. Then $L(A, C) = \{a^{2n+1} \mid n \in \mathbb{N}\}$, a regular language. But with t_1, t_2, t_3 the three transitions of A, from left to right, $\mathsf{Run}(A)\restriction_C = \{t_1^n t_2 t_3^n \mid n \in \mathbb{N}\}$, a nonregular language.

As $\mathsf{Run}(A)$ is effectively obtainable from A, we need only focus on the decidability of the regularity of $\mathsf{Run}(A)\restriction_C$. Note that "moving a cycle" within a run affects neither its being an accepting path, nor its Parikh image. Repeatedly moving the cycles to the leftmost position in the run at which they can occur will be a key ingredient in the following proof. This operation, in particular, will allow to convert the language of runs in an ε-automaton to a set of *bounded* languages, with the property that a path is in $\mathsf{Run}(A)\restriction_C$ iff the repeated moving of cycles leads to a path in one of the bounded languages.

Theorem 2. *It is decidable whether the language of an UnCA is regular.*

Proof. Let (A, C) be a UnCA with $A = (Q, \Sigma, \delta, q_0, F)$. Thanks to Lemma 4, we need only show the decidability of the regularity of $R = \mathsf{Run}(A)\restriction_C$.

We first formalize the discussion made before this theorem. In the following, we use Latin letters b, u, v, w to denote *paths*, and more generally words over δ, as we no longer consider words over Σ. We use the term *cycle* for nonempty paths starting and ending in the same state and with no other state appearing twice, i.e., an elementary cycle in the underlying multigraph. Fix an ordering on the cycles of A: $\{b_1, b_2, \ldots, b_\ell\} \subseteq \delta^*$. Let S be the set of initial paths in A, including the empty path. For $w \in S$, define $\mathsf{States}(w)$ as the set of states visited by w. We see the empty path as from and to q_0, so that $\mathsf{States}(\varepsilon) = \{q_0\}$ and $\mathsf{From}(\varepsilon) = \mathsf{To}(\varepsilon) = q_0$. Define $\alpha\colon S \to (S \times \mathbb{N}^\ell)$ by $\alpha(\varepsilon) = (\varepsilon, \mathbf{0})$ and, for $u \cdot t \in S$ where $t \in \delta$ and $\alpha(u) = (v, \boldsymbol{x})$:

$$\alpha(u \cdot t) = \begin{cases} (v', \boldsymbol{x} + \boldsymbol{e_i}) & \text{if } v \cdot t = v' b_i \wedge \mathsf{States}(b_i) \subseteq \mathsf{States}(v') \ , \\ (v \cdot t, \boldsymbol{x}) & \text{otherwise} \ . \end{cases}$$

Note that α is well-defined and that, for any $u \in S$, $\alpha(u) = (w, \boldsymbol{x})$ is such that w is indeed in S.

In words, applying α removes most of the cycles in a path, and counts them. Hence, if we see $\alpha(u) = (w, \boldsymbol{x})$ as the path w in which b_i is placed x_i times on the first occurrence of $\mathsf{From}(b_i)$ in w, we may interpret the action of α as "moving to the left" each cycle read, while "removing their nesting." Additionally, this path is in R iff u is in R.

In order to make the preceding intuition formal, we define the different bounded languages that represent $\mathsf{Run}(A)$ when the cycles are moved to the leftmost position where they fit. First, for $q \in Q$, fix a compatible ordering on the cycles with q as their origin: $\{b_{(q,1)}, b_{(q,2)}, \ldots, b_{(q,\ell_q)}\}$, i.e., if $b_i = b_{(q,i')}$, $b_j = b_{(q,j')}$, and $i < j$ then $i' < j'$. We write, as usual, $\boldsymbol{b_q}$ for $(b_{(q,1)}, b_{(q,2)}, \ldots, b_{(q,\ell_q)})$. Define, for $q \in Q$, the regular language $B_q = b^*_{(q,1)} b^*_{(q,2)} \cdots b^*_{(q,\ell_q)}$. Now for $w \in S$, let $(q_0, q_1, \ldots, q_n)$ be an ordering of $\mathsf{States}(w)$ such that if q_i is first met before q_j in w, then $i < j$ — that is, the q_i's are ordered in their order of first appearance in w. Further, let $1 = i_0, i_1, \ldots, i_n$ be the positions in w of the first appearance of $q_0, q_1, \ldots, q_n$, respectively. Then we define the bounded regular language $E_w \subseteq S$ by $E_w = B_{q_0} \cdot w_{[i_0, i_1 - 1]} \cdot B_{q_1} \cdot w_{[i_1, i_2 - 1]} \cdots B_{q_n} \cdot w_{[i_n, |w|]}$, where $w = w_1 w_2 \cdots w_{|w|}$, $w_{[a,b]} = w_a w_{a+1} \cdots w_b$. In particular, $E_\varepsilon = B_{q_0}$. Let C_w

be the set $\mathsf{Iter}_{(\boldsymbol{b_{q_0}}, w_{[i_0,i_1-1]}, \ldots, \boldsymbol{b_{q_n}}, w_{[i_n,|w|]})}(E_w \cap R)$ and define I_w using C_w and focusing on the cycles, i.e., for $\boldsymbol{x} \in \mathbb{N}^\ell$, $\boldsymbol{x} \in I_w$ iff:

$$(\boldsymbol{x_{q_0}}, 1, \boldsymbol{x_{q_1}}, 1, \ldots, \boldsymbol{x_{q_n}}, 1) \in C_w \wedge (\forall q \in Q \setminus \{q_0, q_1, \ldots, q_n\})[\boldsymbol{x_q} = \boldsymbol{0}] \ ,$$

where $\boldsymbol{x_q} \in \mathbb{N}^{\ell_q}$, and $x_{(q,i)}$ is understood as the variable x_j for which $b_j = b_{(q,i)}$. Note that if $I_w \neq \emptyset$, then $w \in \mathsf{Run}(A)$. We are now ready to clarify the informal discussion made before the theorem:

Claim 2. For all $u \in S$, $u \in R$ iff $\alpha(u) \in \{(w, \boldsymbol{x}) \mid \boldsymbol{x} \in I_w\}$.

If R is regular, then any $E_w \cap R$ is regular. We will show, using the previous claim as a decision procedure for R, that if all the $E_w \cap R$ are regular, then R is regular. The function α gives a hint of an automaton for R; however, the "accepting set" of Claim 2 clearly establishes that the state set is infinite. To circumvent this problem, we show that we can consider only finite objects with the two following claims, the second being a consequence of Lemma 2.

Claim 3. There is a computable finite set S^{fin} such that any word w appearing as $\alpha(u) = (w, \cdot)$ is in S^{fin}.

Claim 4. Suppose that for all $w \in S^{\text{fin}}$, $E_w \cap R$ is regular. There exist $s \geq 0$, $p \geq 1$ such that for any $\boldsymbol{x} \in \mathbb{N}^\ell$, $\boldsymbol{x} \in I_w$ iff $[\boldsymbol{x}]_{s,p} \subseteq I_w$.

Suppose that for all $w \in S^{\text{fin}}$, $E_w \cap R$ is regular, and let s, p be given by Claim 4. We define a deterministic automaton B for R by:

$$\begin{aligned} B &= (S^{\text{fin}} \times (\mathbb{N}^{|\delta|}/{\equiv_{s,p}}), \quad \delta, \quad \Delta, \quad (\varepsilon, [\boldsymbol{0}]_{s,p}), \quad T) \ , \\ \Delta &= \{(u, [\boldsymbol{x}]_{s,p}) \bullet\!\!-t\!\!\to (u', [\boldsymbol{x} + \boldsymbol{e}]_{s,p}) \mid u \cdot t \in S \wedge \alpha(u \cdot t) = (u', \boldsymbol{e})\} \ , \\ T &= \{(w, [\boldsymbol{x}]_{s,p}) \mid [\boldsymbol{x}]_{s,p} \subseteq I_w\} \ . \end{aligned}$$

The set Δ is well-defined as $\boldsymbol{x} \equiv_{s,p} \boldsymbol{x'}$ implies $\boldsymbol{x} + \boldsymbol{e} \equiv_{s,p} \boldsymbol{x'} + \boldsymbol{e}$. Also, for any word $u \in S$ (and only for them) there is a path from the initial state labeled u.

Claim 5. Suppose that for all $w \in S^{\text{fin}}$, $E_w \cap R$ is regular. Let $u \in S$, $\alpha(u) = (w, \boldsymbol{x})$, and Π be the initial path on B labeled u. Then $\mathsf{To}(\Pi) = (w, [\boldsymbol{x}]_{s,p})$.

Let $u \in S$ and $\alpha(u) = (w, \boldsymbol{x})$. Then $u \in L(B)$ iff, by the Claim 5, $(w, [\boldsymbol{x}]_{s,p}) \in T$, that is, iff $[\boldsymbol{x}]_{s,p} \subseteq I_w$. By Claim 4, this is the case iff $\boldsymbol{x} \in I_w$. By Claim 2, this is the case iff $u \in R \cap S$, i.e., iff $u \in R$. Thus $L(B) = R$ and R is regular.

We now conclude the proof of Theorem 2. As R is regular iff all the $E_w \cap R$ are regular, for $w \in S^{\text{fin}}$, it is sufficient to check whether the latter part is true. Now, for $w \in S^{\text{fin}}$, we can construct a CA for $E_w \cap R$ and we know a socle of $E_w \cap R$ (as we know a socle for E_w); hence Lemma 3 allows to check whether $E_w \cap R$ is regular. □

A DetCA is an UnCA; moreover, DetCA are effectively equivalent [9] to deterministic extended finite automata over $(\mathbb{Z}^k, +, \boldsymbol{0})$ (defined in [10]). Thus:

Corollary 1. *Given a DetCA or an extended finite automaton over $(\mathbb{Z}^k, +, \boldsymbol{0})$, it is decidable whether its language is regular.*

5 A Deterministic Form of UnCA

We present a *deterministic* model equivalent to UnCA. This model is a restriction of the affine Parikh automaton [3] and can be seen as a simple register automaton. As a result of independent interest, we show that CA are equivalent to the nondeterministic variant of this model, and that a seemingly more powerful model (so-called *finite-monoid affine Parikh automata* [2]) is in fact equivalent to CA (resp. UnCA) in its nondeterministic (resp. deterministic) form.

Definition 4 (Affine Parikh automaton [3]). *An* affine Parikh automaton (APA) *of dimension d is a triple (A, U, C) where A is an automaton with transition set δ, $U\colon \delta^* \to \mathcal{F}_d$ is a morphism, and $C \subseteq \mathbb{N}^d$ is semilinear. Its language is $L(A, U, C) = \mu_A(\{\pi \in \mathsf{Run}(A) \mid U_\pi(\mathbf{0}) \in C\})$. The APA is said to be:*

- Deterministic (DetAPA) *if A is deterministic;*
- Finite-monoid (FM-APA, FM-DetAPA) *[2] if $\mathcal{M}(U)$ is finite;*
- Moving (M-APA, M-DetAPA) *if for all $t \in \delta$, $U_t = (M, \boldsymbol{v})$ is such that M is a 0-1-matrix with exactly one* 1 *per row.*

We consider only FM- and M-(Det)APA in the present work. We write $\mathcal{L}_{\text{FM-APA}}$, $\mathcal{L}_{\text{FM-DetAPA}}$, $\mathcal{L}_{\text{M-APA}}$, and $\mathcal{L}_{\text{M-DetAPA}}$ for the classes of languages recognized by FM-APA, FM-DetAPA, M-APA, and M-DetAPA respectively.

Remark 3. An M-(Det)APA of dimension d can be seen as a finite-state *(deterministic) register automaton* with d registers $r_1, r_2, \ldots, r_d$: each transition performs actions of the type $r_i \leftarrow r_{j_i} + k_i$, with $k_i \in \mathbb{N}$, $1 \leq j_i \leq d$, for $1 \leq i \leq d$, and the device accepts iff the underlying automaton accepts and the values of the registers at the end of the computation belong to a prescribed semilinear set.

Theorem 3. $\mathcal{L}_{\text{CA}} = \mathcal{L}_{\text{M-APA}} = \mathcal{L}_{\text{FM-APA}}$.

Proof. We only show $\mathcal{L}_{\text{FM-APA}} \subseteq \mathcal{L}_{\text{CA}}$. Let (A, U, C) be an FM-APA, where $A = (Q, \Sigma, \delta, q_0, F)$. For $t \in \delta$, we write $U_t = (M_t, \boldsymbol{v_t})$, and for $t_1 t_2 \cdots t_n \in \delta^+$, we let $M_{t_1 t_2 \cdots t_n} = M_{t_n} \cdots M_{t_2} \cdot M_{t_1}$. As it is consistent to do, we set $M_\varepsilon = Id$, the identity matrix. We show that $L(A, U, C)$ can be expressed as the union of the languages of a finite number of CA, and that those CA are unambiguous if A is deterministic. We work in 3 steps. (1.) We devise a finite set of automata and show that they recognize the runs π on A while "knowing" M_π (Claim 6). (2.) We show that this extra knowledge allows for the extraction of $U_\pi(\mathbf{0})$ when π is read (Claim 7). We design a semilinear set to constrain this extracted value by C. (3.) We conclude that replacing the labels t of those CA by $\mu_A(t)$ gives a finite set of CA recognizing $L(A, U, C)$.

Step 1: Automata for the Paths of A. The simplest way to construct an automaton for $\mathsf{Run}(A)$ is by replacing the label of each transition t of A by t itself, i.e., we obtain the automaton $(Q, \delta, \Delta, q_0, F)$ where $t = q \bullet\!\!-a\!\!\to q' \in \delta \Leftrightarrow q \bullet\!\!-t\!\!\to q' \in \Delta$. This is the first idea of the present construction. The second idea is that we want, when in a state q, all the possible M_π's for π accepted from q to be the same. Write $\mathcal{M} = \mathcal{M}(U)$. We define, for $q \in Q$ and $M \in \mathcal{M}$,

$B^{\to(q,M)} = (Q \times \mathcal{M},\ \delta,\ \Delta,\ (q,M),\ F \times \{M_\varepsilon\})$, where $\Delta = \{(q,M) \bullet\!\!-t\!\!\to (q',M') \mid t = q \bullet\!\!-\mu(t)\!\!\to q' \in \delta \wedge M'.M_t = M\}$.

It is important to note that even if A is deterministic, $B^{\to(q,M)}$ may not be deterministic. Indeed, let Z be the all-zero matrix, and suppose that, for some $t \in \delta$, $M_t = Z$. Then *any* matrix M' verifies $M'.M_t = Z$, thus from the state $(\mathsf{From}(t), Z)$ there is a transition labeled t to *any* state $(\mathsf{To}(t), M')$ for $M' \in \mathcal{M}$. We now show that these automata indeed recognize the paths π in A, while "knowing" M_π. In order to produce a simple statement, write $A^{\to q}$ for A where the initial state is set to q, then:

Claim 6. For any $q \in Q$ and $M \in \mathcal{M}$, $L(B^{\to(q,M)}) = \{\pi \in \mathsf{Run}(A^{\to q}) \mid M_\pi = M\}$. In particular, $\mathsf{Run}(A) = \bigcup_{M \in \mathcal{M}} L(B^{\to(q_0,M)})$.

Step 2: Retrieving $U_\pi(\mathbf{0})$. In this step, we argue that our previous construction helps in retrieving the value of $U_\pi(\mathbf{0})$ when π is read over some $B^{\to(q,M)}$. The main ingredient is the following simple property: for $t \in \delta$ and $\rho \in \delta^*$, $U_{t\rho}(\mathbf{0}) = M_\rho.\boldsymbol{v_t} + U_\rho(\mathbf{0})$. We now show a property on paths *over* $B^{\to(q,M)}$. First, identify Δ with $\{T_1, T_2, \ldots, T_n\}$, and each T_i with $(q_i, M_i) \bullet\!\!-t_i\!\!\to (q_i', M_i')$; next, write μ_B for the μ function of one of the $B^{\to(q,M)}$'s — this morphism does not depend on the choice of (q,M). Then:

Claim 7. For any $q \in Q$, $M \in \mathcal{M}$, and $\Pi \in \mathsf{Run}(B^{\to(q,M)})$, we have $U_{\mu_B(\Pi)}(\mathbf{0}) = \sum_{i=1}^{n} |\Pi|_{T_i} \times (M_i'.\boldsymbol{v_{t_i}})$.

Now define $C' \subseteq \mathbb{N}^n$ by $(x_1, x_2, \ldots, x_n) \in C' \Leftrightarrow (\sum_{i=1}^{n} x_i \times (M_i'.\boldsymbol{v_{t_i}})) \in C$. Claim 6 and Claim 7 imply that, for $q \in Q$ and $M \in \mathcal{M}$, $L(B^{\to(q,M)}, C') = \{\pi \in \mathsf{Run}(A^{\to q}) \mid M_\pi = M \wedge U_\pi(\mathbf{0}) \in C\}$.

Step 3: from Paths to their Labels. For $q \in Q$ and $M \in \mathcal{M}$, define $D^{\to(q,M)}$ to be the automaton $B^{\to(q,M)}$ where a transition labeled t in $B^{\to(q,M)}$ is relabeled $\mu_A(t)$ in $D^{\to(q,M)}$. Then $L(D^{\to(q,M)}, C') = \mu_A(L(B^{\to(q,M)}, C'))$. Since $\mathsf{Run}(A) = \bigcup_{M \in \mathcal{M}} B^{\to(q_0,M)}$, this implies that $L(A,U,C) = \bigcup_{M \in \mathcal{M}} L(D^{\to(q_0,M)}, C')$. As $\mathcal{M}$ is finite by hypothesis, $L(A,U,C)$ is the finite union of CA languages. The closure of $\mathcal{L}_{\mathrm{CA}}$ under union [9] implies that $L(A,U,C) \in \mathcal{L}_{\mathrm{CA}}$. □

Theorem 4. $\mathcal{L}_{\mathrm{UnCA}} = \mathcal{L}_{\text{M-DetAPA}} = \mathcal{L}_{\text{FM-DetAPA}}$.

Proof (sketch). $\mathcal{L}_{\mathrm{UnCA}} \subseteq \mathcal{L}_{\text{M-DetAPA}}$ is shown in [2, Lemma 5]; $\mathcal{L}_{\text{M-DetAPA}} \subseteq \mathcal{L}_{\text{FM-DetAPA}}$ is immediate.

For $\mathcal{L}_{\text{FM-DetAPA}} \subseteq \mathcal{L}_{\mathrm{UnCA}}$, we simply add a step to the proof of the inclusion $\mathcal{L}_{\text{FM-APA}} \subseteq \mathcal{L}_{\mathrm{CA}}$ of Theorem 3. We note, using the same notations, that if A is deterministic, then for any $q \in Q$ and $M \in \mathcal{M}$, $D^{\to(q,M)}$ is unambiguous. $\mathcal{L}_{\mathrm{UnCA}}$ being closed under union (Proposition 1) this proves the inclusion. □

Remark 4. Theorems 3 and 4 are *effective*, in the sense that one can go from one model to another following an algorithm. This implies in particular, from Theorem 2 that regularity is decidable for FM-DetAPA; we note that it is not decidable for DetAPA [2], which describes a class of languages strictly larger than that of UnCA though expected to be incomparable with that of CA.

6 Conclusion

We showed that $\mathcal{L}_{\mathrm{UnCA}}$ is a class of languages that is closed under the Boolean operations, reversal, and right and left quotient, and that provably fails to be closed under concatenation with a regular language, length-preserving morphisms, and starring. Further, the following problems are decidable for $\mathcal{L}_{\mathrm{UnCA}}$: emptiness, universality, finiteness, inclusion, and regularity. Deciding regularity for UnCA and DetCA is our main result.

We propose three future research avenues. First, the properties of UnCA indicate its suitability for model-checking, and we could envisage real-world applications of verification using UnCA. Second, we translated unambiguous CA to a natural model of *deterministic* register automata; the close inspection of this translation can lead to further advances in our understanding of unambiguity, in particular in the open problems dealing with unambiguous finite automata [4]. Third, we note that the closure properties of $\mathcal{L}_{\mathrm{UnCA}}$ imply that this class can be described by a natural algebraic object (see [1]). This will certainly help in linking UnCA to a first-order logic framework, and thus to some Boolean circuit classes. Hence we hope that UnCA can shed a new light on the classes of circuit complexity.

Acknowledgement. We thank Andreas Krebs for stimulating discussions and comments concerning this work and the anonymous referees their careful reading. The first author thanks Benno Salwey and Dave Touchette for comments on early versions of this paper.

References

1. Behle, C., Krebs, A., Reifferscheid, S.: Typed Monoids – An Eilenberg-Like Theorem for Non Regular Languages. In: Winkler, F. (ed.) CAI 2011. LNCS, vol. 6742, pp. 97–114. Springer, Heidelberg (2011)
2. Cadilhac, M., Finkel, A., McKenzie, P.: Bounded Parikh automata. In: WORDS, pp. 93–102 (2011)
3. Cadilhac, M., Finkel, A., McKenzie, P.: On the expressiveness of Parikh automata and related models. In: NCMA, pp. 103–119 (2011)
4. Colcombet, T.: Forms of determinism for automata. In: STACS, pp. 1–23 (2012)
5. Ginsburg, S., Spanier, E.: Semigroups, Presburger formulas and languages. Pacific Journal of Mathematics 16(2), 285–296 (1966)
6. Ginsburg, S., Spanier, E.: Bounded ALGOL-like languages (1964)
7. Ginsburg, S., Spanier, E.H.: Bounded regular sets. Proceedings of the American Mathematical Society 17(5), 1043–1049 (1966)
8. Ibarra, O.H.: Reversal-bounded multicounter machines and their decision problems. J. ACM 25(1), 116–133 (1978)
9. Klaedtke, F., Rueß, H.: Monadic Second-Order Logics with Cardinalities. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, pp. 681–696. Springer, Heidelberg (2003)
10. Mitrana, V., Stiebe, R.: Extended finite automata over groups. Discrete Appl. Math. 108(3), 287–300 (2001)
11. Parikh, R.J.: On context-free languages. Journal of the ACM 13(4), 570–581 (1966)
12. Sakarovitch, J.: Elements of Automata Theory. Cambridge University Press (2009)