History-deterministic Vector Addition Systems

Sougata Bose
University of Liverpool, UK

David Purser
University of Liverpool, UK

Patrick Totzke
University of Liverpool, UK

Abstract

We consider history-determinism, a restricted form of non-determinism, for Vector Addition Systems with States (VASS) when used as acceptors to recognise languages of finite words, both with coverability and reachability acceptance. History-determinism requires that the non-deterministic choices can be resolved on-the-fly; based on the past and without jeopardising acceptance of any possible continuation of the input word.

Our results show that the history-deterministic (HD) VASS sit strictly between deterministic and non-deterministic VASS regardless of the number of counters. We compare the relative expressiveness of HD systems, and closure-properties of the induced language classes, with coverability and reachability semantics, with and without ε-labelled transitions.

Whereas in dimension 1, inclusion and regularity remain decidable, from dimension two onwards, HD-VASS with suitable resolver strategies, are essentially able to simulate 2-counter Minsky machines, leading to several undecidability results: It is undecidable whether an VASS is history-deterministic, or if a language equivalent history-deterministic VASS exists. Checking language inclusion between history-deterministic 2-VASS is also undecidable.

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

Keywords and phrases Vector Addition Systems, History-determinism, Good-for Games

1 Introduction

Vector addition systems with states (VASSs) are an established model of concurrency with extensive applications in modelling and analysis of hardware, software, chemical, biological and business processes. They are non-deterministic finite automata equipped with a fixed number of integer counters that may be incremented or decremented when changing control state, as long as they remain non-negative.

We explore the notion of *history-determinism* for VASSs when used as acceptors to define languages of finite words. History-determinism is a restricted form of non-determinism. In a nutshell, a non-deterministic automaton is history-deterministic (HD) if there exists a *resolver*, a strategy to stepwise produce a run for any input word given one letter at a time, in such a way that if the given word is in the language of the automaton (some accepting run exists) then the run produced by the resolver is also accepting.

The original motivation for HDness comes from formal verification: most modelling formalisms incorporate some form of non-determinism, e.g., to over-approximate deterministic algorithms, to state specifications concisely, or to model system behaviour due to uncontrollable external environments. However, for non-deterministic models, many formal analysis techniques require costly determinization steps that are often the main barrier to efficient procedures. History-deterministic automata provide a middle ground: they are typically more succinct, or even more expressive, than their deterministic counterparts while preserving some of their good algorithmic properties, They were also called “good-for-games” as they preserve the winner of games under composition and thus allow solving games without determinization.

Any resolver must always chose language-maximal successors. When considering languages of finite words, being able to continue making language-maximal choices is even a sufficient condition for being a resolver. In this case therefore, resolvers can be assumed to be positional (base decision only on the current configuration, not the full history leading to it). Perhaps surprisingly, resolvers for VASSs are not necessarily monotone, and may require more than just comparing counter values to integer thresholds (see Appendix A).

Related Work. VASSs, also known as Petri nets or partially blind counter automata, have been studied intensively since their inception in the 1960s. Early works focussed on modelling capabilities, relative expressiveness and closure properties of their recognised languages [14, 12, 36, 21] but the bulk of research on VASSs concerns decidability and complexity of decision problems [23, 28, 32, 24, 20, 22, 2, 27, 9]. In order to define languages with VASSs, different definitions distinguish between coverability and reachability acceptance conditions, and whether or not silent (ε) transitions are permitted. Checking language emptiness amounts to testing coverability or reachability, which are EXPSPACE [32, 28] and Ackerman-complete [9] respectively. Many other decision problems are undecidable, such as checking language inclusion, bisimulation and related equivalences [19] as well as checking (language) regularity [22]. Universality is undecidable for reachability acceptance [36] and decidable for coverability acceptance, via a well-quasi-order argument but with extremely high complexity (Hyper-Ackermannian in general [20] and still Ackermannian in dimension 1 [18]). These negative results by and large rely on the presence of non-deterministic choice, which motivates restricted forms of non-determinism such as bounded ambiguity (that allows for decidable inclusion [8]) or the notion of history-determinism studied here.

VASS recognisable languages over infinite words are significantly more complex than their finite-word cousins, both topologically and in terms of decision problems: already 1-VASS with (cover) Büchi acceptance can recognise Σ^1_1-complete languages [34, 11] and have an undecidable universality problem [1]. Again, the added complexity is due to non-determinism (languages of deterministic models are Borel, lower in the analytical hierarchy).

History-determinism was introduced independently, with slightly different definitions, by Henzinger and Piterman [16] for solving games without determinization, by Colcombet [7] for cost-functions, and by Kupferman, Safra, and Vardi [25] for recognising derived tree languages of word automata. These different definitions all coincide for finite automata [3] but not necessarily for more general quantitative automata [4].

Until now, history-determinism has mainly been studied for finite-state systems. In this paper we continue a recent line of work [13, 26, 10, 15, 6, 31] that studies the notion for infinite-state models capable of recognising languages beyond (ω-)regular ones. For infinite-state systems, deterministic models are in general less expressive, not just less succinct, than their non-deterministic counterparts. In some cases they can be determinised, such is the case for quantitative automata [4] and timed automata with safety and reachability acceptance [15]. In contrast, for pushdown automata [13] and Parikh automata (VASS with $\mathbb{Z}$-valued counters; [10]), and timed automata with co-Büchi acceptance, allowing history-determinism strictly increases expressiveness (and adds more closure properties) compared to the deterministic variant. Whenever HD automata are strictly less expressive than fully non-deterministic ones, one can reasonably ask if there exists an equivalent HD automaton for a given non-deterministic one. This language HDness question is undecidable for pushdown and Parikh-automata [13, 10]. In fact, even checking if a given (pushdown or Parikh) automaton is itself HD is undecidable (for Parikh automata this follows for example by the undecidability of 2-dim. robot games [30]). On the other hand, checking HDness for timed automata is decidable [15] and various models of quantitative automata [5].

Most closely related to our work is that of Prakash and Thejaswini [31] who study history-deterministic one counter automata (OCA; PDA with unary stack alphabet) and nets (OCN; 1-dimensional VASSs) with state-based (coverability) acceptance. They show that checking automata HDness and inclusion are undecidable for OCA but remain decidable for OCNs. A useful consequence of their construction is that for any OCN one can construct a language equivalent deterministic OCA (with zero-test), albeit with a doubly exponential blow-up. They do not consider closure properties and leave open whether history-deterministic OCNs can be determinised, are equally expressive as fully non-deterministic OCNs, or fall strictly in between in expressiveness. Our work extends and generalises this paper in several directions.

Our Contributions. We study history-deterministic VASSs on finite words and without restricting the dimension. We consider coverability and reachability acceptance conditions, with and without silent (ε) transitions, and in all cases study the relative expressiveness, closure properties, and related decision problems.

We show that HD VASSs are more expressive than deterministic, but less expressive than non-deterministic ones. The same is true for languages recognised by VASSs of any fixed dimensions k, which answers the open question in [31] for $k = 1$. In particular, we provide examples of 1-dim. HD VASSs for which no equivalent deterministic ones exist in any dimension k, and also demonstrate that HD VASSs are strictly more expressive than finitely sequential ones (another restricted form of non-determinism).

We show that HD VASS languages are closed under inverse homomorphisms and intersections for both coverability and reachability semantics, although sometimes necessarily increasing the dimension. Coverability languages are closed under unions, whereas reachability languages are not. Neither are closed under other standard operations, including complementation, concatenation, homomorphisms, iteration and commutative closures.

We report that HDness is not sufficient for decidability of inclusion checking, even for 2-dimensional VASSs. A direct consequence is the undecidability of checking HDness of a given 2-VASS, contrasting decidability in dimension 1. Further, it is undecidable to check if a given VASS has a HD equivalent, and also if a given HD VASS recognises a regular language.

2 Definitions

Vector-Addition Systems and their recognised languages. A k-dimensional *vector-addition system* (k-VASS) is a non-deterministic finite automaton whose transitions manipulate k non-negative integer counters. It is given by $\mathcal{A} = (\Sigma, Q, \delta, s_0, F)$ consisting of a finite alphabet Σ; a finite set of control states Q; a transition relation $\delta \subseteq Q \times \Sigma \cup \{\varepsilon\} \times \mathbb{Z}^k \times Q$; an initial state s_0, and a subset $F \subseteq Q$ of final states. For a transition $t = (s, a, e, s') \in \delta$ we sometimes write $\mathit{label}(t) \stackrel{def}{=} a$ for the letter from $\Sigma \cup \{\varepsilon\}$ it reads and $\mathit{effect}(t) \stackrel{def}{=} e$ for its *effect* on the counters. $\|\delta\|$ denotes the largest absolute effect among all transitions on any counter.

A VASS naturally induces an infinite-state labelled transition system in which each *configuration* is a pair $(s, v) \in Q \times \mathbb{N}^k$ comprising a control state and a *non-negative* integer vector. Every transition $t = (s, a, e, s') \in \delta$ gives rise to steps $(s, v) \xrightarrow{t} (s', v')$ for all $v, v' \in \mathbb{N}$ with $v' = v + e$. We will call a path $\rho = (s_0, v_0) \xrightarrow{t_1} (s_1, v_1) \xrightarrow{t_1} \ldots \xrightarrow{t_k} (s_k, v_k)$ a *run* of the VASS and say it is *cycle* if $s_0 = s_k$. Its *effect* is the sum of all transition effects $\mathit{effect}(\rho) \stackrel{def}{=} \sum_{i=1}^{k} \mathit{effect}(t_i)$. A run ρ as above *reads* the word $\mathit{label}(\rho) = \mathit{label}(t_1)\mathit{label}(t_2)\ldots\mathit{label}(t_k) \in \Sigma^*$. It is *accepting* if it ends in a final configuration.

We consider two different definitions for what constitutes a final (also *accepting*) configurations: In the *coverability* semantics, the set of final configurations is $F \times \mathbb{N}$. In the *reachability* semantics, only configurations from $F \times \mathbf{0}$ are final. We define the *language*

$\mathcal{L}_{\mathcal{A}}(s, v) \subseteq \Sigma^*$ of a configuration (s, v) to contain exactly all words read by some accepting run starting in (s, v) (we omit the subscript $\mathcal{A}$ if the VASS is clear from context). For notational convenience, we will lift this to sets $S \subseteq Q \times \mathbb{N}^k$ of configurations in the natural way: $\mathcal{L}_{\mathcal{A}}(S) \stackrel{def}{=} \bigcup_{(s,v)\in S} \mathcal{L}_{\mathcal{A}}(s, v)$ and define the language of $\mathcal{A}$ as that of its initial state with all counters zero: $\mathcal{L}(\mathcal{A}) \stackrel{def}{=} \mathcal{L}_{\mathcal{A}}(s_0, \mathbf{0})$.

We will sometimes denote languages using short-hand "counting expressions". For instance, we write $a^n b^{\leq n}$ for the language $\{a^n b^m \mid n \geq m\}$ over $\Sigma = \{a, b\}$.

Deterministic and finitely-sequential VASSs. A VASS $\mathcal{A} = (\Sigma, Q, \delta, s_0, F)$ is called ε-free if no transition is labelled by ε. It is *deterministic* if it is ε-free and for every pair $(s, a) \in Q \times \Sigma$ there is at most one transition $t = (s, a, e, s') \in \delta$. A VASS is *finitely sequential* if it is the finite union of deterministic VASSs. That is, all transitions from its initial state s_0 are labelled by ε and lead to an initial state of one of finitely many deterministic VASSs.

History-deterministic VASSs. A VASS is *history-deterministic* if one can resolve non-deterministic choices on-the-fly. More formally, consider a function $r : (Q \times \mathbb{N}^k \times \delta)^*(Q \times \mathbb{N}) \times \Sigma \to \delta$ that, given a finite run $\rho_i = (s_0, v_0) \xrightarrow{t_1} (s_1, v_1) \xrightarrow{t_2} \dots \xrightarrow{t_i} (s_i, v_i)$ and a next letter $a_{i+1} \in \Sigma$, returns a transition $r(\rho_i, a_i) = t_{i+1} = (s_i, e_i, s_{i+1}) \in \delta$ with $label(t_{i+1}) = a_{i+1}$ and $v_i + effect(t_{i+1}) \in \mathbb{N}^k$. This yields, for every word $w = a_0 a_1 \ldots \in \Sigma^*$ and initial configuration (s_0, v_0), a unique run in which the ith step $(s_{i-1}, v_{i-1}) \xrightarrow{t_i} (s_{i+1}, v_{i+1})$ results from a transition chosen by r. Such a function is called *resolver* if for any input word $w \in \mathcal{L}_{\mathcal{A}}(s_0, v_0)$ the constructed run ρ from initial configuration (s_0, v_0) is accepting. A k-VASS is *history-deterministic* if such a resolver exists.

Language Classes. We denote by k-$\mathcal{D}$, k-$\mathcal{H}$,and k-$\mathcal{N}$ the classes of languages recognised by k-dimensional ε-free deterministic, history-deterministic, and fully non-deterministic VASSs, in the coverability semantics. Similarly, let k-$\mathcal{D}^0$, k-$\mathcal{H}^0$, and k-$\mathcal{N}^0$ denote the classes of languages recognised by k-dimensional ε-free deterministic, history-deterministic, and fully non-deterministic VASSs, in the reachability semantics. Finally, define k-$\mathcal{H}_\varepsilon$,k-$\mathcal{N}_\varepsilon$ k-$\mathcal{H}^0_\varepsilon$, and k-$\mathcal{N}^0_\varepsilon$, as above but without the restriction to ε-free systems. When dropping the parameter k we refer to the union over all dimensions k. For instance, $\mathcal{H} \stackrel{def}{=} \bigcup_{k\in\mathbb{N}} k$-$\mathcal{H}$.

3 Expressiveness

We consider the hierarchy of language classes recognised by vector addition systems, varying definitions in three directions: the degree of non-determinism, reachability vs coverability acceptance, and with/without ε-transitions.

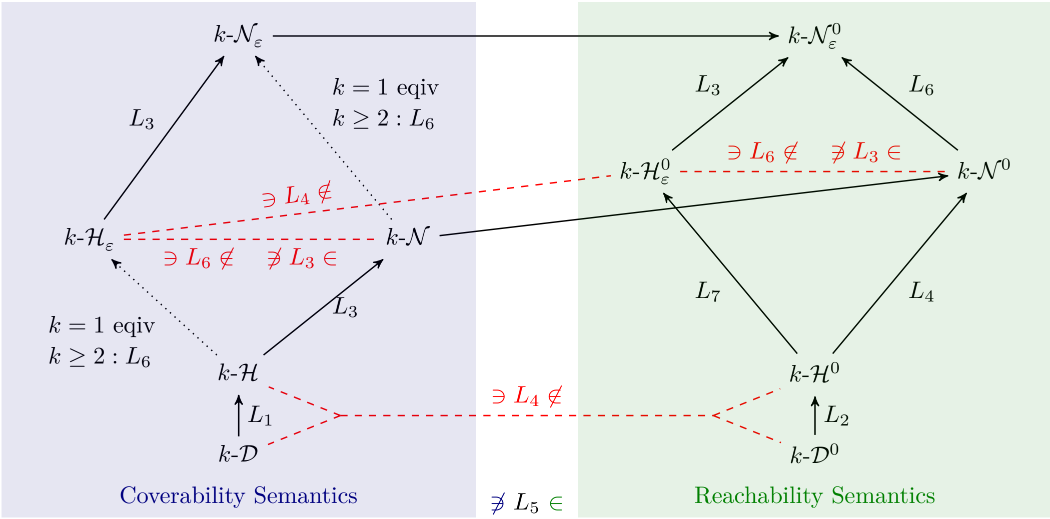
The situation is depicted in Figure 1. We start by looking at the classes defined by ε-free systems (in Section 3.1) before discussing the effect of ε-transitions (in Section 3.2) and following this up with a comparison with finitely-sequential VASS (in Section 3.3).

3.1 Separating determinism, history-determinism and non-determinism

In terms of the classes of languages they define, history-deterministic VASSs are strictly more expressive than deterministic ones, and in turn strictly subsumed by fully non-deterministic ones. The following theorem states this formally. Its proof is split into Lemmas 2–5.

▶ **Theorem 1.** *For all $k \geq 1$, we have k-$\mathcal{D} \subsetneq k$-$\mathcal{H} \subsetneq k$-$\mathcal{N}$ and k-$\mathcal{D}^0 \subsetneq k$-$\mathcal{H}^0 \subsetneq k$-$\mathcal{N}^0$.*

▶ **Lemma 2.** *$L_1 \stackrel{def}{=} a^n b^{\leq n} + a^* b^* c \in 1$-$\mathcal{H} \setminus \mathcal{D}$.*



Language	Definition	Alphabet	Page
L_1	$a^n b^{\leq n} + a^* b^* c$	$\{a, b, c\}$	4
L_2	$a^n b^{\geq n} \#$	$\{a, b, \#\}$	5
L_3	$(a + b)^* a^n b^{\leq n}$	$\{a, b\}$	6
L_4	$a^n b^{\leq n}$	$\{a, b\}$	7
L_5	$a^n b^n$	$\{a, b\}$	7
L_6	$bin(n)\#0^{\leq n}\#$, where $bin(n)$ is n in binary.	$\{0, 1, \#\}$	7
L_7	$a^n b^{\leq n} \#$	$\{a, b, \#\}$	7

Figure 1 Comparison of expressive power of VASS and H-VASS language classes, with and without silent transitions, in reachability and coverability semantics. A solid arrow A → B indicates strict inclusion A ⊊ B, with a separating language denoted on the edge. A red/dashed line indicates pair-wise incomparability, with the separating languages denoted. Dotted arrows indicate a special case.

Proof. L_1 can be recognised by the 1-H-VASS depicted in Figure 2a. Note that the VASS is HD: the only non-deterministic choice is whether to go to q_2 or q_3 on b, for which the resolver must always chose q_2 if available (if the counter is non-zero). The choice of resolver is unique as going to q_3 unnecessarily is not language maximal.

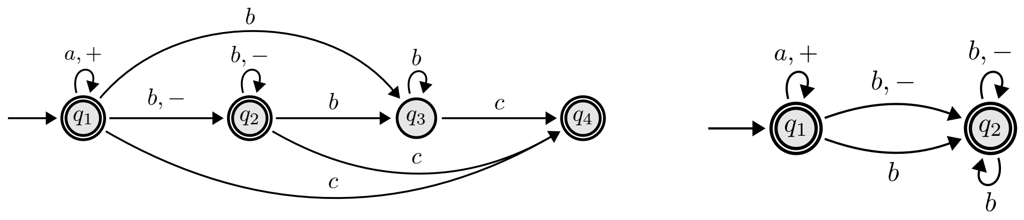
For a contradiction, suppose L_1 accepted by a k-D-VASS with n states. Since $w_{n+1} = a^{n+1}b^{n+1} \in L_1$ the run is accepted. Since there exists $i < j$ such that $a^{n+1}b^i$ is in state q with counter vector $v \in \mathbb{N}^k$ and $a^{n+1}b^j$ is in state q with counter vector $v' \in \mathbb{N}^k$. Since $a^{n+1}b^i \in L_1$, state q is accepting.

Suppose $v' - v \geq \mathbf{0}$, then $a^{n+1}b^{i+(j-i)n} \notin L$ is accepted. Therefore there exists a dimension such that $v' - v$ is negative. Hence for some ℓ we have $a^{n+1}b^{i+(j-i)\ell}$ is a dead run. Hence it cannot accept $a^{n+1}b^{i+(j-i)\ell}c \in L_1$. ◀

▶ **Lemma 3.** $L_2 \stackrel{def}{=} a^n b^{\geq n} \in 1\text{-}\mathcal{H}^0 \setminus \mathcal{D}^0$

Proof. L_2 is recognised by the H-VASS^0 depicted in Figure 2b. On b the resolver can choose between decrementing the counter and no effect, the resolver will always chose to decrement whenever the counter is non-zero.

We have $L_2 \notin \mathcal{D}^0$. Suppose a D-VASS^0 with n states exists, consider the run on the word $w_{n+1} = a^{n+1}b^{n+1} \in L_2$. There exists two prefixes of the run in which $a^{n+1}b^i$ and $a^{n+1}b^j$ revisit a state, and so the system is cyclic on states on extension of $a^{n+1}b^i$ with b^*. Thus, in



(a) A 1-H-VASS recognising L_1.

(b) A 1-H-VASS^0 recognising L_2.

Figure 2 Transitions labelled with + increment the counter by 1, and those labelled by − decrement the counter by 1 and otherwise have no effect on the counter.

order to accept $w_{n+1}b^i$ for all i the automaton must visit only accepting states throughout the cycle. Since $a^{n+1}b^i \notin L_2$ the counter must be non-zero, but zero at $u = a^{n+1}b^{i+(j-i)n}$ since $u \in L$, thus the effect of the cycle is decreasing on some counter, there must exist $k > n$ such that the run is dead on $a^{n+1}b^{i+(j-i)k}$. This is a contradiction as $a^{n+1}b^{i+(j-i)k} \in L_2$. ◀

▶ **Lemma 4.** $L_3 \stackrel{def}{=} \{a,b\}^* a^n b^{\leq n} \in 1\text{-}\mathcal{N} \setminus \mathcal{H}$.

Proof. L_3 can be accepted a 1-N-VASS, which non-deterministically guesses the start of the last a^*b^* block and accepts if there are fewer b's than a's.

We show that $L_3 \notin \mathcal{H}$. Suppose for contradiction there is a k-H-VASS with $|Q|$ states, $\|\delta\|$ the largest effect on a counter in any transition and a resolver r.

Consider a sequence of accepted words $w_\ell = w_{\ell-1}a^{m_\ell}b^{m_\ell}$, with w_0 the empty word, where m_ℓ is large enough so that there exist $r_{\ell,1} < r_{\ell,2} \leq m_\ell$, such that the run given by the resolver r on $w_{\ell-1}a^{r_{\ell,1}}$ has configuration (q_ℓ, v_ℓ) and $w_{\ell-1}a^{r_{\ell,2}}$ has (q_ℓ, u_ℓ), with $u_\ell \geq v_\ell$. In other words, whilst reading a^{m_ℓ}, the run encounters a cycle on state q_ℓ which does not strictly decrease any counter value. This occurs due to Dickson's lemma and depends on $|Q|, \|\delta\|, k$ and $m_1, \ldots, m_{\ell-1}$. This gives an inductive way to build words w_ℓ consisting of ℓ blocks of as and bs such that each a-block visits a non-decreasing cycle. We consider the word w_n for $n = 2^k + 1$ and the run ρ on w_n given by the resolver.

Given a vector $v \in \mathbb{N}^k$, we define $\mathit{support}(v) = \{i \mid v_i \neq 0\}$. Since there are n blocks of a in w_n, each of which has a non-decreasing cycle (q_ℓ, u_ℓ) and (q_ℓ, v_ℓ), for $\ell \in \{1, \ldots, n\}$. However, there are $2^k + 1$ possible choices for $\mathit{support}(u_\ell - v_\ell)$. Therefore, there exists $\ell < \ell'$ such that $\mathit{support}(u_\ell - v_\ell) = \mathit{support}(u_{\ell'} - v_{\ell'})$. In other words, there are two a-blocks which have a non-decreasing cycle such that the effect of the cycles have the same support. Let $R \in \mathbb{N}$ be such that $R(u_\ell - v_\ell) \geq u_{\ell'} - v_{\ell'}$, which exists since $\mathit{support}(u_\ell - v_\ell) = \mathit{support}(u_{\ell'} - v_{\ell'})$ and $u_\ell - v_\ell > \mathbf{0}$ and $u_{\ell'} - v_{\ell'} > \mathbf{0}$.

Let u be the word such that $w_{\ell'-1} = w_\ell u$, i.e, the part of between the ℓth b-block and ℓ'th a-block. Consider the word $w' = w_{\ell-1}a^{m_\ell + R(r_{\ell,2} - r_{\ell,1})}b^{m_\ell}ua^{m_{\ell'} - (r_{\ell',2} - r_{\ell',1})}b^{m_{\ell'}}$. The word w' is therefore obtained by adding $R(r_{\ell,2} - r_{\ell,1})$ many a's in the ℓth a-block and removing $(r_{\ell',2} - r_{\ell',1})$ many a's from the ℓ'th a-block. Note that $w' \notin L_3$, since the last block has more b's than a's. We will show that there is an accepting run on w', by modifying the resolver run on w'_ℓ.

Let $\rho_{\ell'}$ be the run on $w_{\ell'}$ given by the resolver r. We consider the run ρ' where we take the cycle between (q_ℓ, v_ℓ) and (q_ℓ, u_ℓ) an additional R times in the ℓ-th a-block, but removes the cycle between $(q_{\ell'}, v_{\ell'})$ and $(q_{\ell'}, u_{\ell'})$. We show that ρ' is a run on w'. To see this, we must verify that no counter drops below zero in ρ'. Note that the runs $\rho_{\ell'}$ and ρ' are the same till the prefix $w_{\ell-1}a^{r_{\ell,2}}$ after which it reaches the configuration (q_ℓ, u_ℓ). Then it does R additional cycles which results in the configuration $(q_\ell, u_\ell + R(u_\ell - v_\ell))$. From this point ρ'

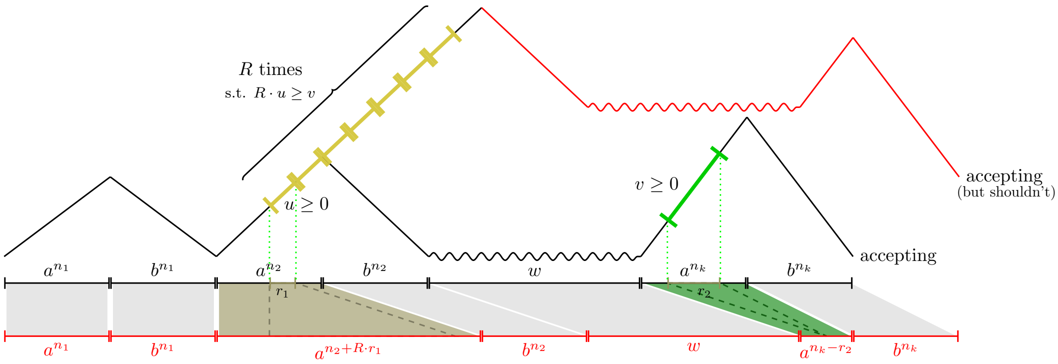


Figure 3 Proof that $L_3 \notin \mathcal{H}$ (Lemma 4). For two cycles of lengths r_1, r_2 chosen in different a^*-blocks with effects $u, v \geq \mathbf{0}$ and $\mathit{support}(u) = \mathit{support}(v)$, repeating the first cycle and removing the second one constructs an accepting run on a word $\notin L_3$.

follows the same sequence of transitions as $\rho_{\ell'}$ till it reads the prefix up to $w_{\ell'-1}a^{r_{\ell',1}}$ ending up in the configuration $(q_{\ell'}, v_{\ell'} + R(u_\ell - v_\ell))$. Since $v_{\ell'} + R(u_\ell - v_\ell) \geq v_{\ell'} + (u_{\ell'} - v_{\ell'}) = u_{\ell'}$, ρ' can follow the suffix of the run $\rho_{\ell'}$ from $(q_{\ell'}, u_{\ell'})$ on $a^{m_{\ell'} - r_{\ell',2}} b^{m_{\ell'}}$, which ends in the same state as $\rho_{\ell'}$ with a non-zero counter value. This is a contradiction as we get a accepting run on $w' \notin L_3$. We conclude that there is no k−H-VASS that recognises the language L_3. ◀

▶ **Lemma 5.** $L_4 \stackrel{def}{=} a^n b^{\leq n} \in 1\text{-}\mathcal{N}^0 \setminus \mathcal{H}^0$

Proof. In the non-deterministic case reachability semantics can recognise $L_4 \in 1\text{-}\mathcal{N}^0$: On a a non-deterministic machine non-determinsitically chooses to increment by 1 or to have no effect, guessing ahead of time how many b's will be seen. On b the machine moves to a new state and counts down, preventing more b's than the guessed number.

However L_4 cannot be recognised with history-determinism. To see this, observe that since $a^n \in L_4$ all the counters must be zero after reading a^n, then for n larger than the number of state the machine cannot distinguish $a^n b^n \in L_4$ and $a^n b^{n+1} \notin L_4$. ◀

3.2 Silent transitions

First observe that $L_5 \stackrel{def}{=} a^n b^n$ can be recognised with reachability semantics (even $\mathcal{D}^0$), but cannot be recognised under coverability semantics (even $\mathcal{N}_\varepsilon$). On the other hand $L_4 = a^n b^{\leq n}$ can be recognised by coverability semantics (even $\mathcal{D}$), but cannot be recognised by $\mathcal{H}^0_\varepsilon$, thus together L_4 and L_5 show pairwise incomparability between reachability and coverability semantics for deterministic and history-deterministic systems. However, if the languages have an end marker then coverability acceptance can be turned into reachability acceptance (with ε-transitions) as ε-transitions can be used to take the counters to zero at the end marker.

The separation between $\mathcal{N}$ and $\mathcal{N}_\varepsilon$ is due to [12] for which $L_6 \stackrel{def}{=} \mathit{bin}(x)\#0^{\leq x}\# \in \mathcal{N}_\varepsilon \setminus \mathcal{N}$, where $\mathit{bin}(n)$ is the binary representation of $n \in \mathbb{N}$ in $1\{0,1\}^*$. This language cannot be recognised without ε transitions (see Appendix B.1 for details). We observe that the same language separates $\mathcal{H}$ and $\mathcal{H}_\varepsilon$, as the 2-VASS of [12] recognising L_6 is in fact history-deterministic. However, in dimension 1, the two classes collapse:

▶ **Lemma 6.** $1\text{-}\mathcal{H} = 1\text{-}\mathcal{H}_\varepsilon$.

While in coverability semantics, the presence of ε-transitions separates languages recognised by k-H-VASS and k-H-VASS$_\varepsilon$ only for dimensions $k \geq 2$, in reachability semantics the separation occurs already in dimension 1: $L_7 \stackrel{def}{=} a^n b^{\leq n}\#$ is in $\mathcal{H}^0_\varepsilon$ but not in $\mathcal{H}^0$.

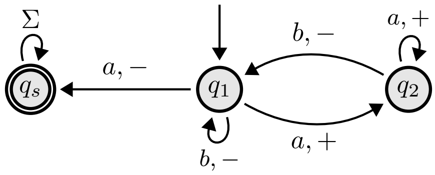


Figure 4 A 1-H-VASS automaton with language $L_8 = \mathcal{L}(q_1, 0)$ that is not finitely sequential. The automaton reads blocks of a's followed by blocks of b's. If some block of a's is followed by fewer b's then the automaton can read anything after the next a. If every block is followed by the same number of a's and b's then it must read another block of the form $a^n b^n$ or $a^n b^{<n}$. The language is thus $L_8 = \bigcup_{k=0}^{\infty} a^{n_0} b^{n_0} \dots a^{n_{k-1}} b^{n_{k-1}} a^{n_k} b^{\leq n_k} a \Sigma^*$.

3.3 Comparison with Finitely Sequential VASS

Recall that finitely sequential VASS are the is a union of finitely many D-VASS. In Lemma 8 we show that language of a finite union of history-deterministic VASS is also history-deterministic. In particular, the deterministic VASSs comprising the finitely sequential VASS are themselves history-deterministic, so any finitely sequential VASS has an equivalent history-deterministic VASS recognising the same language. On the other hand, we show that history-deterministic VASS are strictly more powerful:

▶ **Lemma 7.** *There exists a language in 1-$\mathcal{H}$ that is not finitely sequential.*

Proof. Consider the language $L_8 \overset{def}{=} \mathcal{L}(q_1, 0)$ of the VASS depicted in Figure 4. Observe that it is history-deterministic: when reading a at state q_1, the resolver goes to q_s if possible. This choice is language-maximal and there is no other non-determinism to resolve.

We show the language is not finitely sequential. Suppose for contradiction the language is accepted by a finitely sequential VASS that is the union of k many D-VASS s, each with at most m states. We consider the word $a^{m+1} b^{m+1} \in L_8$. Reading this word, every D-VASS goes through a cycle in the run while reading a^{m+1} and similarly also whilst reading b^{m+1}.

Let $c_1, \dots, c_k$ be the lengths of these cycles while reading a's in each D-VASS respectively, $d_1, \dots, d_k$ be the lengths of the cycles reading b's, and fix $C = \prod_{i \leq k} c_i$ and $D = \prod_{i \leq k} d$. Observe that, for every x, the state of each D-VASS the same state is reached after reading a^{m+1+xC}. Similarly, for any y, the same state is reached after reading $a^{m+1+xC} b^{m+1+yD}$. In particular, fix words $w = a^{m+1+CD} b^{m+1+CD}$ and the words $u = a^{m+1+CD} b^{m+1+(C-1)D}$.

Observe that after reading ua, the system in Figure 4 can be in state q_s and therefore, any extension of ua is accepted. However, the automaton of Figure 4 can only reach state q_2 on wa and so, for any $z \in \mathbb{N}, i \geq 1$, $wa^z b^{z+i} \notin L_8$. Consider this word for $z = m + 1$. Since there is a cycle somewhere while reading b^z, then when reading more b's the automaton visits only states on that cycle. Since $wa^z b^{z+i} \notin L_8$ for $i \geq 1$ either every state on the cycle is non-accepting, or the cycle has a negative effect on at least one counter and therefore becomes unavailable for large enough i.

Recall, in M both wa and ua are in the same control location in each constituent D-VASS, and thus for any $v \in \Sigma^*$ are either wav and uav are in the same control locations (or possibly the run is dead). However, for every z, i, there is some D-VASS in which the word $ua^z b^{z+i}$ is accepting. However, we have argued that for every D-VASS, for sufficiently large i, the run on $wa^z b^{z+i}$ is stuck in a rejecting cycle, or a cycle in which the counter is decreasing. Thus for sufficiently large i, in every D-VASS, either the run on $ua^z b^{z+i}$ is also dead or in a rejecting cycle, which contradicts $ua^z b^{z+i} \in L_8$. ◀

4 Closure Properties

We take a look at closure properties of the classes $\mathcal{H}$ and $\mathcal{H}^0$ recognised by history-deterministic VASSs in coverability and reachability semantics, respectively.

First off, union closure (of $\mathcal{H}$ and $\mathcal{H}^0$) and closure under intersection (for $\mathcal{H}$) can be shown using a straightforward product construction at the cost of increasing the dimension.

▶ **Lemma 8.** *Let $L \in k\text{-}\mathcal{H}$ and $L' \in k'\text{-}\mathcal{H}$. Then $L \cup L' \in (k+k')\text{-}\mathcal{H}$ and $L \cap L' \in (k+k')\text{-}\mathcal{H}$. Let $L \in k\text{-}\mathcal{H}^0$ and $L' \in k'\text{-}\mathcal{H}^0$. Then $L \cap L' \in (k+k')\text{-}\mathcal{H}^0$.*

A naïve product of the two systems recognising L and L' does *not* work for showing the union closure of $\mathcal{H}^0$ because here, acceptance requires all counters to be zero even for inputs that are only in one of the two languages (note the absence of ε-transitions). Indeed, $\mathcal{H}^0$ are not closed under union, as witnessed by $L_9 \stackrel{\text{def}}{=} a^n b^n \cup a^n b^{2n}$ not being in $\mathcal{H}^0$ (See Appendix C).

Taking a direct product yields a H-VASS that may not be optimal in terms of the number of counters and in general, increasing the dimension is not avoidable. For instance, the languages $L_{10} \stackrel{\text{def}}{=} a^n b^{\leq n} c^* \cup a^n b^* c^{\leq n}$ and $L_{11} \stackrel{\text{def}}{=} a^n b^{\leq n} c^* \cap a^n b^* c^{\leq n}$ are not in 1-$\mathcal{H}$, while the individual languages are. Similarly, the language $L_{12} \stackrel{\text{def}}{=} a^n b^n c^* \cap a^n b^* c^n = a^n b^n c^n$ witnesses non-closure of 1-$\mathcal{H}^0$ under intersection.

The theorems below summarise our findings regarding closure properties of history-deterministic classes. Full proofs are in Appendix C.

▶ **Theorem 9.** *$\mathcal{H}$ is closed under union, intersection and inverse homomorphisms. It is not closed under complementation, concatenation, homomorphisms, iteration, nor commutative closure.*

▶ **Theorem 10.** *$\mathcal{H}^0$ is closed under intersection and inverse homomorphisms. It is not closed under union, complementation, concatenation, homomorphisms, iteration, nor commutative closure.*

5 Decision Problems

In this section we consider decision problems related to history-determinism: checking if a given N-VASS is history-deterministic, HD definability (as well s regularity) of its recognised language, and language inclusion between HD VASSs.

Prakash and Thejaswini [31] showed that in dimension 1 (and for coverability semantics), checking HDness and inclusion is decidable in PSPACE by reduction to simulation preorder [17]. This can be generalised slightly as follows.

▶ **Theorem 11.** *Language inclusion $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A})$ is decidable for any 1-H-VASS $\mathcal{A}$ and for any N-VASS $\mathcal{B}$.*

Proof. For that for any 1-H-VASS, one can effectively construct a language equivalent deterministic one-counter automaton (DOCA; a 1-VASS with zero-testing transitions). This is Theorem 19 in [31]. DOCA can be complemented [35] and so the inclusion question is equivalent to the emptiness (reachability) of $\overline{\mathcal{A}} \times \mathcal{B}$, a VASS with one zero-testable counter, which is decidable [33]. ◀

We continue to show that in higher dimensions, these questions are undecidable. Our constructions proving this are similar, yet require subtle differences, and are all based on weakly simulating two-counter machines [29]. Let us recall these in a suitable syntax first.

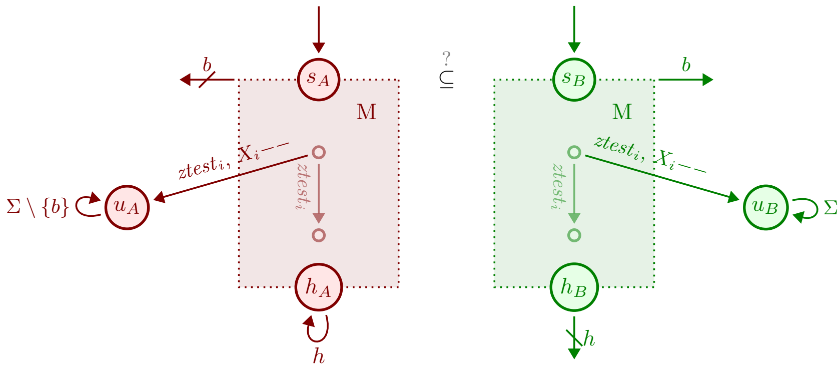


Figure 5 The 2-VASSs A (in red) and B (in green) both include a copy of, and weakly simulate, a given 2CM M. For any zero-testing operation ztest_i in M both can go to a sink state if counter i is in fact non-zero, reading the letter ztest_i and decreasing the VASS counter i, as indicated by the effect vector $X_i{-}{-}$. The extra letter b ensures that $\mathsf{B} \not\subseteq \mathsf{A}$; Only A can accept words that consist of valid sequences of 2CM operations and that end in the letter h.

▶ **Definition 12.** *A two-counter Minsky machine (2CM) $M = (Q, q_0, q_h, \delta)$ consists of is a finite set Q, including a distinguished starting and final state q_0, q_h, respectively, as well as a finite set of transitions $\delta \subseteq Q \times \Gamma \times Q$, where $\Gamma = \{inc_1, inc_2, dec_1, dec_2, ztest_1, ztest_2\}$ are the operations on the counters*[1]*.*

A configuration of M is an element of $Q \times \mathbb{N}^2$, comprising the current state and the value of the two counters. For every state q either: **1.** *There is only one transition of the form (q, inc_i, q'). This allows to move from state q to q', increment counter i by one and leaves the other counter untouched; or* **2.** *There are exactly two transitions from q, of the form $(q, ztest_i, q')$ and (q, dec_i, q''). The former allows to move to q' without changing the counters, but only if counter i has value 0. The latter allows to move from q to q'' and decrease counter i, and leaves the other counter unchanged.*

Notice that from any configuration there is exactly one possible successor configuration. We can therefore speak of *the* run of M, and its sequence of counter operations, from the initial configuration $(s_0, 0, 0)$. We say that M *terminates* if its run visits the final state q_h. W.l.o.g., we can assume that whenever M terminates then with both counters at value 0.

Deciding whether a given 2CM terminates is undecidable [29]. An easy consequence, and the bases for our construction for regularity, is the undecidability of checking finiteness of the reachability set for a given 2CM.

▶ **Lemma 13.** *It is undecidable to check, for given 2CM M, if its run visits infinitely many different configurations.*

5.1 Checking HDness and Inclusion

We focus on the questions whether a given VASS is history-deterministic, and whether language inclusion holds for two languages given by H-VASS. For languages of finite words these two decision problems are intrinsically linked (see Appendix A).

[1] Readers may be more familiar with an instruction of the form if $C_i = 0$ goto q_ℓ else goto q_k, this can be simulated by a ztest_i to q_ℓ and a decrement followed by an increment to q_k.

▶ **Lemma 14.** *For a given 2CM M one can construct two history-deterministic 2-VASSs with initial states s_A and s_B, respectively, so that $\mathcal{L}(s_A, 0) \subseteq \mathcal{L}(s_B, 0)$ if, and only if, the unique valid run of M never reaches a halting state.*

Proof. Suppose we are given 2CM M with designated initial and halting states s and h, respectively, and let Γ denote the set of counter operations. W.l.o.g., there is exactly one valid sequence of counter operations that is either infinite or finite. We define two 2-VASS A and B over the alphabet $\Sigma = \Gamma \uplus \{b, h\}$. These are just copies of, and just weakly simulate the machine M: For every state q of M, there are states q_A and q_B; For every transition $q \xrightarrow{\gamma} q'$ of M, there are corresponding edges $q_A \xrightarrow{\gamma} q'_A$ and $q_B \xrightarrow{\gamma} q'_B$ that read the letter γ and manipulates the counter accordingly: if $\gamma = \text{inc}_i$ (or dec_i) then counter i is incremented (or decremented,respectively). If $\gamma = \text{ztest}_i$ then counter i remains as is. The only accepting states so far are h_A and h_B, corresponding to the designated halting state of M.

Additionally, for every zero-testing transition $q \xrightarrow{\text{ztest}_i} q'$ in M, both A and B have a transition from state q that decreases counter i and goes to a new, accepting, sink state u with language $\supseteq (\Gamma \cup \{h\})^*$. This way, both systems will accept any word that prescribes a run of M that contains a "counter cheat", meaning that the word contains operation ztest_i but the run of M so far ends in a configuration where counter i is not zero.

We now modify the systems A and B so that they differ in two ways:

1. the halting state h_A of A admits a h-labelled step (to itself) but s_B does not.
2. All states in B have b-labelled steps (to the accepting sink u_B) but none of As states do.

See Figure 5 for a depiction of the constructed 2-VASS.

Notice that $\mathcal{L}(B) \not\subseteq \mathcal{L}(A)$ by design, because no word containing the letter b can be accepted by A. Notice that both A and B are indeed history-deterministic: they only choices to be resolved are upon reading a zero-testing latter ztest_i from a configuration where the corresponding counter i is not zero. In any such case, moving to the sink is language maximal.

It remains to argue that $\mathcal{L}(s_A) \not\subseteq \mathcal{L}(s_B)$ if, and only if, M has a finite run from initial configuration to its final state. Indeed, if M terminates via a sequence $\rho = e_0 e_1, \ldots e_k$, then $\mathcal{L}(A)$ contains the word $\rho \cdot h$. Since this run does not contain "cheats" nor letters b, the system B cannot possibly reach the winning sink u_B and therefore not accept. Conversely, if M does not terminate, then any word $\rho \in \Gamma^* \cdot h$ accepted by A must prescribe a run of M that contains a cheat. Say $\rho = \rho_1 \cdot \text{ztest}_i \cdot \rho_1 \cdot h$. But then, B will be able to reach the sink u_B after reading he prefix $\rho_1 \cdot \text{ztest}_i$ and thus accept. ◀

The construction in the previous lemma works both in coverability and reachability semantics (note that we assume that a 2CM terminates with counters 0). The next two theorems are direct consequences and again hold for coverability and reachability semantics.

▶ **Theorem 15.** *Checking language inclusion is undecidable for 2-HD VASSs.*

▶ **Theorem 16.** *It is undecidable to check if a given 2-VASS is history-deterministic.*

Proof. By reduction from 2CM termination: Construct the two systems A and B as given by Lemma 14 and add one new initial state s that, upon reading some letter b can move to the initial state s_A of A or s_B of B. By Proposition 21, the so-constructed system is HD iff $\mathcal{L}(s_A) \subseteq \mathcal{L}(s_B)$, which is true iff M does not terminate. ◀

5.2 Checking HDness of VASS Languages

We turn to showing undecidability of *language* history-determinism, i.e., the question if for a given VASS there exists an equivalent history-deterministic VASS. We start with the more interesting and involved case, for the coverability semantics (Theorem 17) and present an easier construction for reachability (Theorem 18) afterwards.

We give a proof by reduction from the 2CM halting problem, combining the constructions to show the non-HDness of $L_3 = (a,b)^* a^n b^{\leq n}$, (Lemma 4) and the proof of [22] that checking regularity for N-VASS languages is undecidable.

▶ **Theorem 17.** *It is undecidable to check if $\mathcal{L}(\mathcal{A}) \in \mathcal{H}$ holds for a given N-VASS $\mathcal{A}$.*

Proof. By reduction from the 2CM halting problem. For a given 2CM M with states Q_M and counter operations $\Gamma = \{\text{inc}_1, \text{inc}_2, \text{dec}_1, \text{dec}_2, \text{ztest}_1, \text{ztest}_2\}$ we construct a 3-VASS $\mathcal{A} = (\Sigma, Q, \delta, s_0, F)$ so that $\mathcal{L}(\mathcal{A})$ is history-deterministic iff the faithful run of M is finite.

We refer to the three counters as X_1, X_2, X_3 and write $X_i{-}{-}$ and $X_i{+}{+}$ for the effects of (VASS) transitions that decrement/increment counter i only.

The construction. $\mathcal{A}$ uses the alphabet $\Sigma = \Gamma \cup \{a, b\}$, consisting of counter operations of M and two fresh symbols. The control states of $\mathcal{A}$ mimic those of M, except that in between any simulated step of M, $\mathcal{A}$ can read a word in a^+b^+: For every state $q \in Q_M$ we introduce states q_{in}, q_{out} and q_{step}. In addition, we add three other states $sink, r_1, r_2$. We make $sink$ universal by adding self-loops $(s, a, \mathbf{0}, s)$ for every letter $a \in \Sigma$. First we consider the simulation of M.

For every step $q \xrightarrow{\gamma} p$ of M, $\mathcal{A}$ has a transition $t = (q_{out}, \gamma, e, p_{in})$ from q_{out} to p_{in} that reads the letter $label(t) = \gamma$ and manipulates the counter accordingly: if $\gamma = \text{inc}_i$ then $e = X_i{+}{+}$; if $\gamma = \text{dec}_i$ then $e = X_i{-}{-}$; if $\gamma = \text{ztest}_i$ then $e = \mathbf{0}$. In addition, for zero-testing steps $q \xrightarrow{\text{ztest}}_i p$, $\mathcal{A}$ in M, $\mathcal{A}$ contains a decreasing transition $t = (q_{out}, \text{ztest}_i, X_i{-}{-}, sink)$ to the universal sink state. From a state q_{in}. There are two possible continuations:

1. Reading a word in a^+b^+ and moving to q_{out}, via transitions $q_{in} \xrightarrow{a,\mathbf{0}} q_{step}$, $q_{step} \xrightarrow{a,\mathbf{0}} q_{step}$, $q_{step} \xrightarrow{b,\mathbf{0}} q_{out}$ and $q_{out} \xrightarrow{b,\mathbf{0}} q_{out}$.
2. Reading a word in $a^n b^{\leq n}$ and stopping. For this, there are transitions $q_{in} \xrightarrow{a,X_3++} r_1$, $r_1 \xrightarrow{a,X_3++} r_1$, $r_1 \xrightarrow{b,X_3--} r_2$ and $r_2 \xrightarrow{b,X_3--} r_2$.

The accepting states of $\mathcal{A}$ are $F = \{r_2, sink\}$. Its initial state is $s_0 = q_{out}$, where $q \in Q_M$ is the initial state of M.

The recognised language of the constructed 3-VASS $\mathcal{A}$ contains sequences of instructions of M interspersed with blocks of the form a^+b^+. Let's call a sequence $\gamma_1\gamma_2 \ldots \gamma_k \in \Gamma^*$ of operations in M *faithful* if for all $i \leq k$, γ_i is the ith instruction in the run of M from its initial configuration $(q, 0, 0)$. Clearly, for any k less or equal to the length of the run of M, there is a unique faithful sequence ρ_k of length k. Define $\mathsf{Correct}_k \stackrel{def}{=} \gamma_1(a^+b^+)\gamma_2(a^+b^+)\gamma_3 \ldots (a^+b^+)\gamma_k$ where $\gamma_1\gamma_2 \ldots \gamma_k = \rho_k$. Let $\mathsf{Incorrect}_k \subseteq \Sigma^*$ contain exactly all words $w\gamma \in \Sigma^* \setminus \mathsf{Correct}_k$ where $w \in \mathsf{Correct}_{k-1}$ and $\gamma \in \{\text{ztest}_1, \text{ztest}_2\}$. That is, words whose projection into the operations of M is faithful up to step $k-1$ but that contain an incorrect zero-test at step k.

Observe that if the faithful sequence of length k takes M to (q, C_1, C_2) then $\mathcal{A}$ can read any word in $\mathsf{Correct}_k$ and every run on such a word leads to the configuration $(q_{in}, C_1, C_2, 0)$. Such a run of $\mathcal{A}$ can be extended in two ways to reach an accepting state. Either by reading a word in $a^n b^{\leq n}$ to reach r_2, or by continuing on the run of M and eventually erroneously reading a ztest$_i$ to reach *sink*. We can therefore write the language of $\mathcal{A}$ as

$$\mathcal{L}(\mathcal{A}) \quad = \quad \bigcup_{k\geq 0} \mathsf{Correct}_k \cdot (a^n b^{\leq n}) \quad \cup \quad \bigcup_{k\geq 0} \mathsf{Incorrect}_k \cdot \Sigma^*$$

HDness. We show that if M terminates, meaning its run has some length $k \in \mathbb{N}$, then $\mathcal{L}(\mathcal{A})$ is history-deterministic. Observe that for every $0 \leq i \leq k$, both languages $\mathsf{Correct}_i$ and $\mathsf{Incorrect}_i$ are regular. We can concatenate a DFA recognising the former with a 1-H-VASS for $a^n b^{\leq n}$ to construct an 1-H-VASS recognising $\mathsf{Correct}_i \cdot (a^n b^{\leq n})$. Now, $\mathcal{L}(\mathcal{A})$ is the finite union of k many 1-H-VASS languages and therefore recognisable by a k-dimensional H-VASS.

It remains to show that if the run of M is infinite, then $\mathcal{L}(\mathcal{A})$ is not in k-$\mathcal{H}$, for any k. Our proof mirrors the proof of Lemma 4, except that we interleave $\{a, b\}$-blocks with the faithful operations of M. Suppose towards a contradiction that there exists a k-H-VASS $\mathcal{B}$ with states $Q_\mathcal{B}$ and let $\rho = \gamma_1\gamma_2, \dots \in \Gamma^\omega$ denote the infinite run of M. That is, every length-i prefix ρ_i is faithful. Consider a sequence $(w_n)_{n\geq 0}$ of words in $\mathcal{L}_\mathcal{B}()$ such that $w_0 = \varepsilon$ and otherwise $w_\ell = w_{\ell-1}\gamma_\ell a^{m_\ell} b^{m_\ell}$ with m_ℓ large enough so that the resolved run on w_ℓ contains a non-decreasing cycle while reading the last a-block. Say,

$$(s_0, \mathbf{0}) \xrightarrow{w_{\ell-1}\gamma_\ell a^{r_{\ell,1}}} (q_\ell, u_\ell) \xrightarrow{a^{r_{\ell,2}}} (q_\ell, v_\ell)$$

with $u_\ell \leq v_\ell$. This is well-defined by Dickson's Lemma.

Setting $n = |Q_\mathcal{B}|2^k + 1$ is sufficiently high so that there must be $\ell < \ell'$ with $q_\ell = q_{\ell'}$ and $\mathit{support}(u_\ell - v_\ell) = \mathit{support}(u_{\ell'} - v_{\ell'})$. Take R be such that $R(u_\ell - v_\ell) \geq (u_{\ell'} - v_{\ell'})$ and let u be the word such that $w_{\ell'-1} = w_\ell u$. Now consider the word

$$w' = w_{\ell-1}\gamma_\ell a^{m_\ell + R(r_{2,\ell})} b^{m_\ell} u \gamma_{\ell'} a^{m_{\ell'} - r_{2,\ell'}} b^{m_{\ell'}}$$

that results from w_n by removing one iteration of the loop in block ℓ' and making up for it by inserting R iterations of the loop in block ℓ. Notice that w' is accepted by the run that follows the resolved run on w_n and repeats the designated loops on the extra letters. However, $w' \notin \mathcal{L}(\mathcal{A})$ because its last $\{a, b\}$-block contains more b's than a's. ◀

Notice that if the given 2CM terminates then our construction produces a history-deterministic VASS where the number of counters corresponds to the length of the terminating run. Therefore it remains open whether the language k-HDness problem is decidable, which ask whether there is an equivalent k-HDVASS for the given language.

The analogous statement for reachability is simpler to prove, by reduction from the universality problem, which is undecidable in reachability semantics [36, Theorem 10].

▶ **Theorem 18.** *It is undecidable to check if $\mathcal{L}(\mathcal{A}) \in \mathcal{H}^0$ holds for a given N-VASS $\mathcal{A}$.*

5.3 Regularity

We turn to the decision problem whether a given VASS recognises a regular language. This regularity question is undecidable for general N-VASS [22]. It again turns out that for history-deterministic VASSs, the decidability status of regularity depends on the dimension. For 1-H-VASS, one can effectively construct a language equivalent DOCA [31], for which checking regularity remains decidable [2, 35].

▶ **Theorem 19.** *Given a 1-H-VASS $\mathcal{A}$, checking if $\mathcal{L}(\mathcal{A})$ is regular is decidable in* EXPSPACE.

Although checking regularity of DOCA is NL-complete, the added complexity here is due to the doubly exponentially large DOCA produced in the reduction. We now show undecidability already for dimension 2.

▶ **Theorem 20.** *Given a 2-H-VASS $\mathcal{A}$, it is undecidable if $\mathcal{L}(\mathcal{A})$ is regular.*

Proof. By reduction from the finiteness problem for 2CM (Lemma 13). For a given 2CM M we construct a 2-H-VASS whose language will be regular iff M's run visits only finitely many configurations. We make the argument for coverability semantics first.

Let $\rho = \gamma_1\gamma_2\ldots$ be the faithful run of M and $|\rho| \in \mathbb{N} \cup \{\infty\}$ for its length. Write $\mathit{correct}_k$ for its length-k prefixes and let x_k be 1 plus the sum of both counter-values in the configuration M reaches after reading $\mathit{correct}_k$. Further, wherever $\mathit{correct}_k = \mathit{correct}_{k-1}\mathrm{dec}_i$, define $\mathit{incorrect}_k$ as $\mathit{correct}_{k-1}\mathrm{ztest}_i$.

Consider the language $L = G \uplus B$ over the alphabet $\Sigma = \Gamma \uplus \{a\}$,

$$G \stackrel{def}{=} \bigcup_{k\geq 0} \left(\rho_k \cdot a^{\leq x_k}\right) \qquad \text{and} \qquad B \stackrel{def}{=} \bigcup_{k\geq 0} \left(\mathit{incorrect}_k \cdot \Sigma^*\right)$$

G consists of words that describe some length-k prefix of M's run followed by x_k or fewer symbols a; B contains all words describing the run of M up to length-k, followed by an incorrect zero-test, and then anything.

We claim that this language L is recognised by a 2-H-VASS. To see this, again build a VASS that weakly simulates M as done before, for example in the proof of Theorem 17. This will simulate increment and decrement operations faithfully, reading letters inc_i or dec_i, respectively. For any step $q \xrightarrow{\mathrm{ztest}_i} q'$ in M, the VASS $\mathcal{A}$ will have a transition $(q, \mathrm{ztest}_i, \mathbf{0}, q')$ as well as one that reads ztest_i, decreases counter i and leads to a universal state. This allows to accept exactly all words in B. In addition, from any state q of M, $\mathcal{A}$ can move to a new countdown phase: there is a transition $q \xrightarrow{a,\mathbf{0}} c$ to a new, final, control state that can continue to read $a's$ while at least one of the counters remains non-zero. This allows to accept exactly all words in G. Note that the only non-determinism is for letters ztest_i when M's ith counter after reading ρ_i is not zero. In this case, the only language-maximal choice is to move to the universal state. The constructed system is therefore history-deterministic.

To conclude the proof, we argue that L is regular iff ρ visits only finitely many configurations. Indeed, if so, then G is finite because all x_i, $i \leq k$ are bounded, and B is regular because at most k many words $\mathit{incorrect}_k$ exist. So L is the finite union of regular languages and thus regular.

Conversely, suppose that M's run ρ visits infinitely many different configurations. Then in particular, there are infinitely many faithful prefixes ρ_k. Let us assume towards contradiction that L is regular and recognised by a DFA with d many states. We pick a prefix ρ_k so that $x_k > d$ and consider the word $\rho_k a^{x_k} \in L$. While reading the suffix a^m, our DFA must repeat some cycle of length $c \leq d$. But then it must also accept $\rho_k a^{x_k+c} \notin L$ by going through that cycle twice.

The same proof goes through for the reachability semantics if we set $G \stackrel{def}{=} \bigcup_{k\geq 0} (\rho_k \cdot a^{=x_k})$ and $B \stackrel{def}{=} \bigcup_{k\geq 0} \left(\mathit{incorrect}_k \cdot \Sigma^{\geq x_k - 1}\right)$. Then again, if the run of M visits finitely many configurations then both G and B are regular. Otherwise G is not regular. The extra symbols at the end (of words in G and B) allow a run of the VASS $\mathcal{A}$ to decrease the counters to 0 and accept (and therefore to conclude that language $L = G \uplus B$ is in 2-$\mathcal{H}^0$). ◀

References

1 Stanislav Böhm, Stefan Göller, Simon Halfon, and Piotr Hofman. On büchi one-counter automata. In *International Symposium on Theoretical Aspects of Computer Science*, volume 66 of *LIPIcs*, pages 14:1–14:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.STACS.2017.14.

2 Stanislav Böhm, Stefan Göller, and Petr Jancar. Bisimulation equivalence and regularity for real-time one-counter automata. *Journal of Computer and System Sciences*, 80(4):720–743, 2014. doi:10.1016/j.jcss.2013.11.003.

3 Udi Boker and Karoliina Lehtinen. Good for games automata: From nondeterminism to alternation. In *International Conference on Concurrency Theory*, volume 140 of *LIPIcs*, pages 19:1–19:16, 2019.

4 Udi Boker and Karoliina Lehtinen. History Determinism vs. Good for Gameness in Quantitative Automata. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 213 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.FSTTCS.2021.38.

5 Udi Boker and Karoliina Lehtinen. Token games and history-deterministic quantitative automata. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 120–139. Springer International Publishing, 2022.

6 Sougata Bose, Thomas A. Henzinger, Karoliina Lehtinen, Sven Schewe, and Patrick Totzke. History-deterministic timed automata are not determinizable. In *International Workshop on Reachability Problems*, 2022.

7 Thomas Colcombet. The theory of stabilisation monoids and regular cost functions. In *International Colloquium on Automata, Languages and Programming*, pages 139–150, 2009.

8 Wojciech Czerwinski and Piotr Hofman. Language inclusion for boundedly-ambiguous vector addition systems is decidable. In *International Conference on Concurrency Theory*, volume 243 of *LIPIcs*, pages 16:1–16:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CONCUR.2022.16.

9 Wojciech Czerwinski and Lukasz Orlikowski. Reachability in vector addition systems is ackermann-complete. In *Annual Symposium on Foundations of Computer Science*, pages 1229–1240. IEEE, 2021. doi:10.1109/FOCS52979.2021.00120.

10 Enzo Erlich, Shibashis Guha, Ismaël Jecker, Karoliina Lehtinen, and Martin Zimmermann. History-deterministic parikh automata, 2022. arXiv:2209.07745.

11 Olivier Finkel and Michal Skrzypczak. On the topological complexity of ω-languages of non-deterministic petri nets. *Information Processing Letters*, 114(5):229–233, 2014. doi:10.1016/j.ipl.2013.12.007.

12 Sheila A. Greibach. Remarks on blind and partially blind one-way multicounter machines. *Theoretical Computer Science*, 7:311–324, 1978. doi:10.1016/0304-3975(78)90020-8.

13 Shibashis Guha, Ismaël Jecker, Karoliina Lehtinen, and Martin Zimmermann. A Bit of Nondeterminism Makes Pushdown Automata Expressive and Succinct. In *International Symposium on Mathematical Foundations of Computer Science*, volume 202 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 53:1–53:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.MFCS.2021.53.

14 Michel Henri Theódore Hack. Petri net language. 1976.

15 Thomas A. Henzinger, Karoliina Lehtinen, and Patrick Totzke. History-deterministic timed automata. In *International Conference on Concurrency Theory*, 2022.

16 Thomas A. Henzinger and Nir Piterman. Solving games without determinization. In *EACSL Annual Conference on Computer Science Logic*, pages 395–410. Springer Berlin Heidelberg, 2006.

17 Piotr Hofman, Slawomir Lasota, Richard Mayr, and Patrick Totzke. Simulation problems over one-counter nets. *Logical Methods in Computer Science*, 12(1), 2016. doi:10.2168/LMCS-12(1:6)2016.

18 Piotr Hofman and Patrick Totzke. Trace inclusion for one-counter nets revisited. *Theoretical Computer Science*, 11174, 2017. doi:10.1016/j.tcs.2017.05.009.

19 Petr Jancar. Nonprimitive recursive complexity and undecidability for petri net equivalences. *Theoretical Computer Science*, 256(1-2):23–30, 2001. doi:10.1016/S0304-3975(00)00100-6.

20 Petr Jancar, Javier Esparza, and Faron Moller. Petri nets and regular processes. *Journal of Computer and System Sciences*, 59(3):476–503, 1999. doi:10.1006/jcss.1999.1643.

21 Matthias Jantzen. Language theory of petri nets. In *Petri Nets: Central Models and Their Properties, Advances in Petri Nets 1986, Part I, Proceedings of an Advanced Course, Bad Honnef, Germany, 8-19 September 1986*, volume 254 of *LNCS*, pages 397–412. Springer, 1986. doi:10.1007/BFb0046847.

22 Petr Jančar and Faron Moller. Checking regular properties of petri nets. In *International Conference on Concurrency Theory*, pages 348–362, 1995.

23 Richard M. Karp and Raymond E. Miller. Parallel program schemata. *Journal of Computer and System Sciences*, 3(2):147–195, 1969. doi:10.1016/S0022-0000(69)80011-5.

24 S. Rao Kosaraju. Decidability of reachability in vector addition systems. In *STOC'82*, pages 267–281, 1982. doi:10.1145/800070.802201.

25 Orna Kupferman, Shmuel Safra, and Moshe Y Vardi. Relating word and tree automata. *Annals of Pure and Applied Logic*, 138(1-3):126–146, 2006.

26 Karoliina Lehtinen and Martin Zimmermann. Good-for-games ω-pushdown automata. *Logical Methods in Computer Science*, 18, 2022.

27 Jérôme Leroux. The reachability problem for petri nets is not primitive recursive. In *Annual Symposium on Foundations of Computer Science*, pages 1241–1252. IEEE, 2021. doi:10.1109/FOCS52979.2021.00121.

28 Richard J. Lipton. The reachability problem requires exponential space. Technical Report 62, Yale University, 1976.

29 Marvin L. Minsky. *Computation: finite and infinite machines*. Prentice-Hall, Inc., 1967.

30 Reino Niskanen, Igor Potapov, and Julien Reichert. Undecidability of two-dimensional robot games. 58:73:1–73:13, 2016. doi:10.4230/LIPIcs.MFCS.2016.73.

31 Aditya Prakash and K. S. Thejaswini. On history-deterministic one-counter nets. In *International Conference on Foundations of Software Science and Computational Structures*. Springer, 2023. doi:10.1007/978-3-031-30829-1_11.

32 Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theoretical Computer Science*, pages 223–231, 1978. doi:10.1016/0304-3975(78)90036-1.

33 Klaus Reinhardt. Reachability in petri nets with inhibitor arcs. *Electronic Notes in Theoretical Computer Science*, 223:239–264, 2008. doi:https://doi.org/10.1016/j.entcs.2008.12.042.

34 Michal Skrzypczak. Büchi VASS recognise w-languages that are sigma^1_1 - complete. *CoRR*, abs/1708.09658, 2017. arXiv:1708.09658.

35 Leslie G. Valiant and Mike Paterson. Deterministic one-counter automata. *Journal of Computer and System Sciences*, 10(3):340–350, 1975. doi:10.1016/S0022-0000(75)80005-5.

36 Rüdiger Valk and Guy Vidal-Naquet. Petri nets and regular languages. *Journal of Computer and System Sciences*, 23(3):299–325, 1981. doi:https://doi.org/10.1016/0022-0000(81)90067-2.

A The Structure of Resolvers

We observe that any resolver r must always make language-maximal choices. To formalise, let us write $Post_a(s,v) \stackrel{def}{=} \{(s',v') \mid (s,v) \xrightarrow{a} (s',v')\}$ for the finite set of possible a-successor configurations of (s,v). Suppose a run produced by r leads up to configuration (s_i,v_i) and for the next letter a_i, it selects a continuation $(s_i,v_i) \xrightarrow{a_i} (s_{i+1},v_{i+1})$. Then $\mathcal{L}(s_{i+1},v_{i+1}) \supseteq \bigcup \mathcal{L}(Post_{a_i}(s_i,v_i))$. When considering languages of finite words a useful observation is that

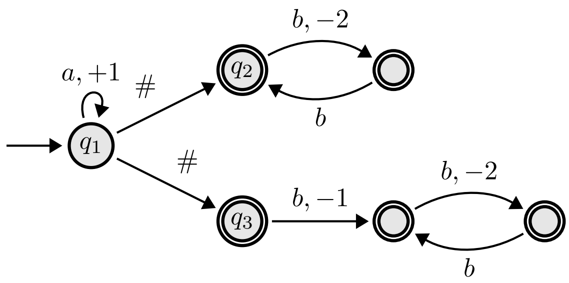


Figure 6 A 1-H-VASS with a resolver that requires a resolver that depends on more than threshold comparisons.

making only language maximal choices is not only necessary, but also a sufficient condition for r to be a resolver.

▶ **Proposition 21.** *A function r as above is a resolver iff all it's choices are language maximal.*

This does not depend on the finiteness of the state space. A direct consequence is that resolvers can be assumed to be *positional*: That is, if any resolver r exists then also one whose decisions only depend on the current configuration and given letter, not on the whole prefix run: $r(\rho(s, v), a) = r(\rho'(s, v), a)$ for any two $\rho, \rho' \in (Q \times \mathbb{N}^k \times \Sigma)^*$ and letter a.

Proof. Suppose a candidate resolver r does not always make language-maximal choices. That is, for some word w the corresponding run chosen by r ends in some configuration c and for some letter a, it moves to a successor configuration c' that is not language maximal. Then there exist a suffix word w' so that some run from c on aw' is accepting but no run from c' on w' is accepting, including the one chosen by r. So r is not a resolver.

Conversely, suppose a candidate resolver r that always makes language maximal choices and assume towards a contradiction that it is not a resolver. This means that Player 1, wins the letter game from the initial configuration c_0: for some word $w = a_0a_1 \dots a_k \in \mathcal{L}(c_0)$ the run $c_0 \xrightarrow{a_0} c_1 \xrightarrow{a_1} \dots \xrightarrow{a_k} c_{k+1}$ constructed by r is not accepting. Since some accepting run on w exists, there must be a last configuration c_j on this run which can still accept the suffix $w[j] = a_ja_{j+1} \dots a_k$. This uses the assumption that we consider languages of finite words, not infinite ones. We conclude that the step $c_j \xrightarrow{a_j} c_{j+1}$ was *not* language maximal, since $w[j] \in \mathcal{L}(c_j)$ but $a_{j+1} \dots a_k \notin \mathcal{L}(c_{j+1})$. Contradiction. ◀

For 1-H-VASS it is sufficient for a resolver to be semi-linear [31], meaning that, for each state and proposed letter, the counter configurations for which each available choice should be chosen can be expressed as a semi-linear set. However [31] does not show the full power of semi-linear resolvers are required, and most natural examples appear to only require threshold queries (and often only to distinguish between zero and non-zero counter). We show that threshold comparisons with the counters is not sufficient: in the following example the system must have access to the parity of the counter.

▶ **Example 22.** Consider the 1-H-VASS depicted in Figure 6. Observe that:

- $\mathcal{L}(q_1, 0) = \{a^n\#b^m \mid m \leq n\}$,
- $\mathcal{L}(q_2, n) = \{b^m \mid m \leq n$ if n even or $m \leq n - 1$ if n odd$\}$,
- $\mathcal{L}(q_3, n) = \{b^m \mid m \leq n$ if n odd or $m \leq n - 1$ if n even$\}$.

Hence, upon reading # a resolver must decide decide whether $\mathcal{L}(q_2, n) \subset \mathcal{L}(q_3, n)$ or $\mathcal{L}(q_3, n) \subset \mathcal{L}(q_2, n)$, which is possible by looking at the parity of the counter value.

B Additional Material for Expressiveness (Section 3)

B.1 Comparison of History-deterministic VASS with and without ε-transitions

Given a number $x \in \mathbb{N}$ let $bin(x)$ be the binary representation of x in $1\{0,1\}^*$. Consider the language $L_6 = bin(n)\#0^{\leq n}\#$, which is, for any number n, the binary representation of n followed by $\#$, followed by at most n-many 0's. A result of [12] says that this language cannot be represented by any real-time machine, that is, machines without ε-transitions. For completeness, we recall the argument in brief: observe that the value of the maximum counter can be at most $\|\delta\| \log(n)$ after reading $bin(n)$, where $\|\delta\|$ is the maximal counter effect of any transition. Therefore there are at most $poly(\log(n))$ configurations reachable after reading $bin(n)$, however, there are $2^{\log(n)-1}$ different numbers of length $|bin(n)|$. As a result, there are two numbers $n < m$ with $|bin(n)| = |bin(m)|$ (with $\log(n)$ large enough) for which reading either $bin(n)$ or $bin(m)$ has the same configuration; in which case either $bin(n)\#0^m$ is incorrectly accepted or $bin(m)\#0^n$ is incorrectly rejected.

▶ **Lemma 6.** $1\text{-}\mathcal{H} = 1\text{-}\mathcal{H}_\varepsilon$.

Proof. Clearly $1\text{-}\mathcal{H} \subseteq 1\text{-}\mathcal{H}_\varepsilon$. We show that ε transitions can be removed from a 1-H-VASS$_\varepsilon$. If there are no cycles then this is done in the standard way, merging them with the prior letter-consuming transitions. For cycles there are three cases. There are finitely many destinations on zero cycles and can be treated as in the acyclic case. Negative cycles are not beneficial, so the resolver should not iterate them. Therefore, we only add transitions necessary to access particular states, but keeping the counter maximal. Cycles with positive effect, for the purposes of maximal language acceptance, should be repeated infinitely. Thus it suffices to go to a copy of the automaton behaving only as a state-machine (without counter effects). Our procedure, adds finitely many counter maximal transitions. We observe the new system is also HD, when reading a letter the resolver can decided where the resolver for the system would go with a and then a sequence of ε transitions and move to a place with the same state and at least as high counter, which is language maximal. ◀

C Additional Material for Closure Properties (Section 4)

▶ **Lemma 8.** *Let $L \in k\text{-}\mathcal{H}$ and $L' \in k'\text{-}\mathcal{H}$. Then $L \cup L' \in (k+k')\text{-}\mathcal{H}$ and $L \cap L' \in (k+k')\text{-}\mathcal{H}$. Let $L \in k\text{-}\mathcal{H}^0$ and $L' \in k'\text{-}\mathcal{H}^0$. Then $L \cap L' \in (k+k')\text{-}\mathcal{H}^0$.*

Proof. Let L and L' be recognised by by k-H-VASS $\mathcal{A}$ and k'-H-VASS $\mathcal{B}$ respectively, in the coverability semantics. W.l.o.g., assume that both are *complete*, meaning there exists a (not necessarily accepting) run on every word. This can be guaranteed by adding new transitions to non-accepting sink states (which any resolver will avoid if possible). The language $L \cup L'$ is accepted by the $(k+k')$-H-VASS obtained by taking product $\mathcal{A} \times \mathcal{B}$, with $k+k'$ counters, where the first k counters simulate the counters of $\mathcal{A}$ and the last k' counters simulate the counters of $\mathcal{B}$. A state in the product (q, q') is accepting if either q or q' is accepting, or both are accepting. The resolver for the product from a configuration $((q_1, q'_1), v)$ on a letter a chooses the product of the transitions chosen by the resolver of $\mathcal{A}$ and $\mathcal{B}$ from the configurations (q_1, v_k) and $(q'_1, v_{k'})$ respectively on the letter a, where v_k and v'_k are the projection of v to the first k and last k' coordinates.

The construction works similarly for the intersection of $\mathcal{H}$ and $\mathcal{H}^0$ languages by taking accepting states (q, q') in the product if both q and q' are accepting. ◀

We now consider closure of the language classes $\mathcal{H}$ and $\mathcal{H}^0$ for other operations, defined here shortly. The *concatenation* of two languages L and L' is the languages $L \cdot L' = \{w = uv \mid u \in L,\ v \in L'\}$. Let Σ and Γ be some alphabets and $h : \Sigma^* \to \Gamma^*$ be a homomorphism. The *homomorphic image* of a language $L \subset \Sigma^*$ is $h(L) = \{h(w) \mid w \in L\} \subset \Gamma^*$. Similarly, the *inverse homomorphic image* of a language $L \subset \Gamma^*$ is $h^{-1}(L) = \{w \in \Sigma^* \mid h(w) \in L\}$.

Let $\Sigma = \{a_1, a_2, \ldots, a_n\}$ be an alphabet. The *Parikh image of a word* $w \in \Sigma^*$ is the vector $\Psi(w) = (v_1, v_2, \ldots, v_n)$, where v_i is the number of occurrences of a_i in w. The *Parikh image of a language* L is the set of Parikh images of words in L. For a language L, its *commutative closure* $CC(L)$ is the language $\{w \mid \exists u \in L.\Psi(w) = \Psi(u)\}$.

▶ **Theorem 9.** *$\mathcal{H}$ is closed under union, intersection and inverse homomorphisms. It is not closed under complementation, concatenation, homomorphisms, iteration, nor commutative closure.*

Proof. The proof for closure under union and intersection is by Lemma 8. For inverse homomorphisms, let $L \subset \Gamma^*$ be in $\mathcal{H}$ accepted by a k-H-VASS A with a resolver r. Let Q be the set of states of A and $\|\delta\|$ be the largest absolute effect among all transitions. Let $h : \Sigma^* \to \Gamma^*$ be a homomorphism and ℓ be such that $|h(a)| \leq \ell$, for all $a \in \Sigma$. Then $h^{-1}(L)$ is accepted by the k-H-VASS A' with the states as $Q \times D^k$, where $D = [0, \ell\|\delta\|]$. For every $a \in \Sigma$, a transition in A' from (q, v) to (q', v') on a will correspond to a run in A' from q to q' on $h(a)$, so that the resolver r' for A' will simply choose the transition corresponding to the run chosen by resolver r in A on $h(a)$.

We need to show that A' does not accept any word $w \notin h_1(L)$. To show this, we need to ensure that if a run on $h(a)$ gets blocked due to some counter dropping below zero, the corresponding transition in A' is also blocked. To do this, the transition in A' has effect equal to the maximum negative effect in any prefix of the run on $h(a)$. The rest of the effect in the run on $h(a)$ is delayed to the next transition. Since the maximum effect is bounded by $\ell\|\delta\|$, this can be stored in the states. The next transition will therefore have the sum of the effect delayed from the previous transition and the maximum negative effect in the prefix of the current transition. The details of the construction are below.

Let $\rho = t_1 t_2 \ldots t_{k'}$ be a path in A from q to q' on $h(a) = \mathit{label}(\rho)$. Let $f_{ij} = \mathit{effect}(t_1 t_2 \ldots t_j)|_i$, i.e, the effect in the prefix up to jth transition projected to the ith counter. Let $f_i = \min_j(f_{ij})$ and $e_i = \min(f_i, 0)$. Thus e_i gives the largest negative effect in any prefix of the run. For every path ρ on $h(a)$, we have a transition $((q, v), a, e', (q', v'))$ in A' if $e' = e + v$, where $e = (e_1, e_2, \ldots, e_k)$ and $v' + e = \mathit{effect}(\rho)$. A state (q, v) of A' is initial if q is initial in A and $v = \mathbf{0}$ and (q, v) in A' is accepting if q is accepting in A.

Now, we give counterexamples for the operations under which $\mathcal{H}$ is not closed.

Complementation. Consider the language $L_4 = \{a^n b^{\leq n}\}$ which is in $\mathcal{D}$. The complement of L_4 is not even in $\mathcal{N}$. Indeed if it were in $\mathcal{N}$, then $L_4^c \cap a^* b^* = a^n b^{\geq n}$ would be in $\mathcal{N}$, which is not the case.

Concatenation. Consider the language $L_3 = \Sigma^* \cdot a^n b^{\leq n}$. By Lemma 4, $L_3 \notin \mathcal{H}$.

Homomorphisms. Consider $L = \{c, d\}^* a^n b^{\leq n} \in \mathcal{H}$ which is accepted by even a D-VASS. Let h be the homomorphism $h(c) = a, h(d) = b$, which gives $h(L) = L_3 \notin \mathcal{H}$ by Lemma 4.

Kleene star. Consider $(a^n b^{\leq n})^*$ which is the Kleene star of $L_4 \in \mathcal{H}$. The proof of Lemma 4, also shows that L_4^* is not in $\mathcal{H}$.

Commutative closure. Consider the commutative closure of L_4, $L = CC(L_4) = \{w \mid \#a \geq \#b\}$. If L is in $\mathcal{H}$, then $L \cap b^* a^* = b^n a^{\geq n}$ is also in $\mathcal{H}$ as $\mathcal{H}$ is closed under intersection. However $b^n a^{\geq n}$ is not even in $\mathcal{N}$. ◀

▶ **Theorem 10.** *$\mathcal{H}^0$ is closed under intersection and inverse homomorphisms. It is not closed under union, complementation, concatenation, homomorphisms, iteration, nor commutative closure.*

Proof. Closure under intersection follows from Lemma 8. For the *inverse homomorphic image*, a construction similar to the $\mathcal{H}$, with states $(q, \mathbf{0})$ taken to be accepting in A' for every q that is accepting in A. Note that any accepting run on $h(w)h(a)$, for any word $w \in \Sigma^*$ and $a \in \Sigma$, the effect of the run on $h(a)$ cannot be positive on any counter as it would lead to a non-zero counter value in the final configuration contradicting that the run is accepting. Therefore, the maximal negative effect encoded in the transition in our construction will always lead to a state $(q, \mathbf{0})$ and not delay any positive effect for later. This proves that $\mathcal{H}^0$ are closed under inverse homomorphic image.

Now, we give counterexamples for the operations under which $\mathcal{H}^0$ is not closed.

Unions. Consider the language $L_9 = a^n b^n \cup a^n b^{2n}$. Both of the languages $a^n b^n$ and $a^n b^{2n}$ are in 1-$\mathcal{H}^0$. Suppose L_9 is recognised by a k-H-VASS^0 A. Since, $a^n b^n$ is in L, the resolver gives an accepting for all n, i.e, in a final state with all counters 0. Let $n_1 < n_2$ be such that the run given by resolver on $a^{n_1} b^{n_1}$ and $a^{n_2} b^{n_2}$ end in the same state q. Since $a^{n_1} b^{n_1+n_1}$ is also accepted, the resolver extends the run from $(q, \mathbf{0})$ on the suffix b^{n_1} and gives an accepting run. This also gives an accepting run on $a^{n_2} b^{n_1+n_2}$ which is a contradiction.

Complementation. Consider $L_2 = a^n b^{\geq n}$ which is in 1-$\mathcal{H}^0$. Recall that $L_4 a^n b^{\leq n} = L_2^c \cap a^* b^*$ is not in $\mathcal{H}^0$ by Lemma 5. If L_2^c was in $\mathcal{H}^0$, then so would L_4 due to closure under intersection leading to a contradiction.

Concatenation. Consider the concatenation of a^* and $a^n b^n$, both in $\mathcal{D}^0$, which gives the language $L_4 = a^n b^{\leq n}$, which is not in $\mathcal{H}^0$ by Lemma 5.

Homomorphisms. Consider $\{c\}^* a^n b^n$ which is in 1-$\mathcal{H}^0$ (even 1-$\mathcal{D}^0$), and $h(c) = a$, gives the language L_4 as above which is not in $\mathcal{H}^0$ by Lemma 5.

Kleene star. Consider $L_{13} = a^n b^n a^*$, which is 1-$\mathcal{H}^0$. Indeed L_{13}^* is not in $\mathcal{H}^0$. The run given by the resolver on $a^n b^n a^m$ must end with $\mathbf{0}$, for all $m \geq 0$. In particular, there exists $m_1 < m_2$ such that the configuration reached on $a^n b^n a^{m_i}$ are the same and therefore accept the same continuations. Thus $a^n b^n a^{m_2} b^{m_2}$ cannot be distinguished from $a^n b^n a^{m_1} b^{m_2}$, which is a contradiction.

Commutative closure. $CC(a^n b^{\geq n}) \cap b^* a^* = b^n a^{\leq n}$, which is not in $\mathcal{H}^0$ by the same proof as Lemma 5 for $a^n b^{\leq n}$ not being in $\mathcal{H}^0$. ◀

To show that taking product for union ($\mathcal{H}$) and intersection ($\mathcal{H}$ and $\mathcal{H}^0$) is not optimal in terms of number of counters, we have the following theorem.

▶ **Theorem 23.** 1-$\mathcal{H}$ *is not closed under union and intersection.* 1-$\mathcal{H}^0$ *is not closed under intersection.*

Proof. Union. Consider the language $L_{10} = a^n b^{\leq n} c^* \cup a^n b^* c^{\leq n}$, which is the union of two languages in 1-$\mathcal{H}$. Suppose L_{10} is in 1-$\mathcal{H}$. Let $|Q|$ be the number of states and $\|(\|\delta)$ be the maximum counter effect of transitions in the 1-H-VASS accepting $a^n b^{\leq n} c^* \cup a^n b^* c^{\leq n}$. Consider the sequence of words $w_n = a^n b^n$‘bc^n and the runs given on these words by the resolver. Let (q_n, v_n) be the configuration reached after reading $a^n b^n$, for every n. Now, we consider two cases. Suppose there exists a bound B such that $v_n < B$ for all n. Then there exists $n_1 < n_2$ such that $(q_{n_1}, v_{n_1}) = (q_{n_2}, v_{n_2})$. Since $bc^{n_2} \in L(q_{n_2}, v_{n_2})$, we get an accepting run on $a^{n_1} b^{n_1} bc^{n_2}$ which is a contradiction. Therefore, the counter values v_n must be unbounded.

Now, consider the infinite sequence of words $w_{n_1}, w_{n_2}, \ldots$ such that the state reached after $a^{n_i}b^{n_i}$ is the same, i.e, $q_{n_1} = q_{n_2} = \ldots$ and the last $|Q|$ many transitions leading to (q_{n_i}, v_{n_i}) are also the same. Note that since there are finitely many choices of the last state and $|Q|$ length sequence of transitions, such an infinite subsequence must exist. Let $(q^j_{n_i}, v^j_{n_i})$ denote the configuration reached in the run on $a^{n_i}b^{n_i}$ given by resolver after the prefix $a^{n_i}b^{n_i-j}$, for $j \leq |Q|$. It is easy to see that $v_{n_1} < v_{n_2} < \ldots$, as $L(q_{n_i}, v_{n_i}) \subsetneq L(q_{n_{i'}}, v_{n_{i'}})$ for $i < i'$ witnessed by $bc^{n_{i'}} \in L(q_{n_{i'}}, v_{n_{i'}})$ but not in $L(q_{n_i}, v_{n_i})$. Since the last $|Q|$ transitions leading to (q_{n_i}, v_{n_i}) are the same, we can also conclude that $v^j_{n_1} < v^j_{n_2} < \ldots$, for all $j \leq |Q|$. We write q^j to denote $q^j_{n_i}$ since the state is the same for all choices of i.

Note that $b^jc^* \subseteq L(q^j, v^j_{n_i}) \not\supseteq b^{j+1}c^*$, for all n_i, $j \leq |Q|$. The inclusion of b^jc^* is immediate because the resolver must make language maximal choices. However, if $b^{j+1}c^*$ is included, then we get an accepting run on $a^{n_i}b^{n_i+1}c^{n_i+i}$, which is a contradiction. This shows that $b^jc^* \subseteq L(q^j, c) \not\supseteq b^{j+1}c^*$ for any $c > v^j_{n_1}$. This is because languages from configurations with the same state are monotone in the value of the counter.

Note that there exists a $j < j'$ such that $q^j = q^{j'}$ as we look at runs whose last $|Q|$ transitions (and therefore $|Q| + 1$ states) are the same. Now choose n_i such that $v^j_{n_i}$ and $v^{j'}_{n_i}$ are both bigger than $min(v^j_{n_1}, v^j_{n_1})$. Therefore, by the previous observation, $b^jc^* \subseteq L(q^{j'}, v^{j'}_{n_i}) \not\supseteq b^{j+1}c^*$. This means $b^{j'}c^*$ is not accepted from $(q^{j'}, v^{j'}_{n_i})$ which contradicts that the run was chosen by a resolver. This concludes the proof that the language L_{10} is not accepted by any 1-H-VASS.

Intersection. Consider the language $L_{11} = a^nb^{\leq n}c^* \cap a^nb^*c^{\leq n}$ and suppose it is accepted by 1-H-VASS. Consider the runs on $a^nb^nc^n$ given by the resolver. If the configuration reached after a^nb^n has counter value bounded, then by a similar reasoning to the union case, we can find $n_1 < n_2$ such that the configuration reached by the resolver after reading $a^{n_1}b^{n_1}$ and $a^{n_2}b^{n_2}$ are the same and we get an accepting run on $a^{n_1}b^{n_1}c^{n_2}$ which is a contradiction. If the configuration is unbounded, we get a n such that the configuration reached after reading a^nb^n has counter value $> (|Q| + 1)\|(\|\delta)$. This allows to repeat a cycle in b^n block as the maximum decreasing effect is at most $(|Q| + 1)\|(\|\delta)$. This gives an accepting run on a^nb^m, where $m > n$, which is a contradiction.

For the reachability semantics, the proof is even simpler as $L_{12} = a^nb^nc^n$ is not even context-free and therefore not definable even with zero tests on the counter. ◀

D Additional Material for Decision Problems (Section 5)

▶ **Lemma 13.** *It is undecidable to check, for given 2CM M, if its run visits infinitely many different configurations.*

Proof. Suppose that one could decide above question. Then one can also decide the halting problem: if the set of reachable configurations is infinite then clearly M does not halt. Otherwise, we can determine if M halts by simulating it either until it halts, or if it re-visits one configuration without halting. ◀

▶ **Theorem 18.** *It is undecidable to check if $\mathcal{L}(\mathcal{A}) \in \mathcal{H}^0$ holds for a given $\mathbb{N}$-VASS $\mathcal{A}$.*

Proof. We reduce from the undecidable universality problem for VASS languages in reachability semantics [36, Theorem 10]. The construction is the same as for the regularity problem of Parikh-automata, recently presented in [10]. For an alphabet Σ let $\Sigma_\$ = \Sigma \uplus \{\$\}$ for some fresh symbol $\$ \notin \Sigma$. For two words u, v let $u \otimes v$ be the word $w = (a_1, b_1)(a_2, b_2)\ldots(a_k, b_k)$ so that either $u = a_1a_2\ldots a_k$ and $b_1b_2\ldots b_k \in v\$^*$ or $v = b_1b_2\ldots b_k$ and $a_1a_2\ldots a_k \in u\$^*$.

For two languages $L, L' \subseteq \Sigma^*$ define their *cross-union* $L \ovee L \subseteq (\Sigma_\$^2)^*$ to be the langugae of words $u \otimes v$ such that $u \in L$ or $v \in L'$. That is, for any word $w \in L \ovee L$, either the projection into the first components is $\pi_1(w) \in L$ or that into the second components $\pi_2(w) \in L'$.

Recall the language $L_4 = a^n b^{\leq n} \in \mathcal{N}^0 \setminus \mathcal{H}^0$, which is not HD recognisable. To show our claim, let L be some given N-VASS language and consider the language $L_{14} \stackrel{def}{=} \$ \cdot (L \ovee \emptyset) \cup \$ \cdot (\emptyset \ovee L_4)$. This is clearly in $\mathcal{N}^0$. Now, if $L = \Sigma^*$ is universal then $L \ovee \emptyset$ is universal over $\Sigma_\$^2$ and so $L_{14} = \$(\Sigma_\$^2)^* \in \mathcal{H}^0$ (even, regular). If conversely, suppose L is not universal as witnessed by $w \notin L$, then L_{14} cannot be recognised by any H-VASS^0 for the same reason as $a^n b^{\geq n} \notin \mathcal{H}^0$: suppose it is accepted by some k-H-VASS^0 run on n states and consider run of the resolver on the word $u = \$(w \otimes a^{|w|+n+1}) \in L_{14}$, thus must end with counter $\mathbf{0}$. The extension of u by $(\$, b)^{n+1}$ is also accepting, it must remain at $\mathbf{0}$ and cycle on accepting states. Hence $u(\$, b)^{|w|+n+1} \in L_{14}$ cannot be distinguished from $u(\$, b)^{|w|+n+2} \notin L_{14}$. ◀

E Index of Languages used in this paper

Name	Definition	Alphabet	Page
L_1	$a^n b^{\leq n} + a^* b^* c$	$\{a, b, c\}$	4
L_2	$a^n b^{\geq n} \#$	$\{a, b, \#\}$	5
L_3	$(a + b)^* a^n b^{\leq n}$	$\{a, b\}$	6
L_4	$a^n b^{\leq n}$	$\{a, b\}$	7
L_5	$a^n b^n$	$\{a, b\}$	7
L_6	$bin(n)\#0^{\leq n}\#$, where $bin(n)$ is n in binary.	$\{0, 1, \#\}$	7
L_7	$a^n b^{\leq n} \#$	$\{a, b, \#\}$	7
L_8	$\bigcup_{k=0}^{\infty} a^{n_0} b^{n_0} \ldots a^{n_{k-1}} b^{n_{k-1}} a^{n_k} b^{\leq n_k} a \Sigma^*$.	$\{a, b\}$	8
L_9	$a^n b^n \cup a^n b^{2n}$	$\{a, b\}$	9
L_{10}	$a^n b^{\leq n} c^* \cup a^n b^* c^{\leq n}$	$\{a, b, c\}$	9
L_{11}	$a^n b^{\leq n} c^n \cap a^n b^* c^{\leq n}$	$\{a, b, c\}$	9
L_{12}	$a^n b^n c^n$	$\{a, b, c\}$	9
L_{13}	$a^n b^n a^* = a^n b^n c^* \cap a^n b^* c^n$	$\{a, b\}$	20