Deciding Polynomial Termination Complexity for VASS Programs

Michal Ajdarów
Masaryk University, Brno, Czech Republic

Antonín Kučera
Masaryk University, Brno, Czech Republic

Abstract

We show that for every fixed degree $k \geq 3$, the problem whether the termination/counter complexity of a given demonic VASS is $\mathcal{O}(n^k)$, $\Omega(n^k)$, and $\Theta(n^k)$ is **coNP**-complete, **NP**-complete, and **DP**-complete, respectively. We also classify the complexity of these problems for $k \leq 2$. This shows that the polynomial-time algorithm designed for strongly connected demonic VASS in previous works cannot be extended to the general case. Then, we prove that the same problems for VASS games are **PSPACE**-complete. Again, we classify the complexity also for $k \leq 2$. Tractable subclasses of demonic VASS and VASS games are obtained by bounding certain structural parameters, which opens the way to applications in program analysis despite the presented lower complexity bounds.

2012 ACM Subject Classification Theory of computation → Models of computation

Keywords and phrases Termination complexity, vector addition systems

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2021.30

Related Version *Full Version*: `https://arxiv.org/abs/2102.06889` [1]

Funding The work is supported by the Czech Science Foundation, Grant No. 21-24711S.

1 Introduction

Vector addition systems with states (VASS) are a generic formalism expressively equivalent to Petri nets. In program analysis, VASS are used to model programs with unbounded integer variables, parameterized systems, etc. Thus, various problems about such systems reduce to the corresponding questions about VASS. This approach's main bottleneck is that interesting questions about VASS tend to have high computational complexity (see, e.g., [8, 15, 16]). Surprisingly, recent results (see below) have revealed computational tractability of problems related to *asymptotic complexity* of VASS computations, allowing to answer questions like "Does the program terminate in time polynomial in n for all inputs of size n?", or "Is the maximal value of a given variable bounded by $\mathcal{O}(n^4)$ for all inputs of size n?". These results are encouraging and may enhance the existing software tools for asymptotic program analysis such as SPEED [11], COSTA [2], RAML [12], Rank [3], Loopus [18, 19], AProVE [10], CoFloCo [9], C4B [7], and others. In this paper, we give a full classification of the computational complexity of deciding polynomial termination/counter complexity for demonic VASS and VASS games, and solve open problems formulated in previous works. Furthermore, we identify structural parameters making the asymptotic VASS analysis computationally hard. Since these parameters are often small in VASS program abstractions, this opens the way to applications in program analysis despite the established lower complexity bounds.

The *termination complexity* of a given VASS $\mathcal{A}$ is a function $\mathcal{L} : \mathbb{N} \to \mathbb{N}_\infty$ assigning to every n the maximal length of a computation initiated in a configuration with all counters initialized to n. Similarly, the *counter complexity* of a given counter c in $\mathcal{A}$ is a function

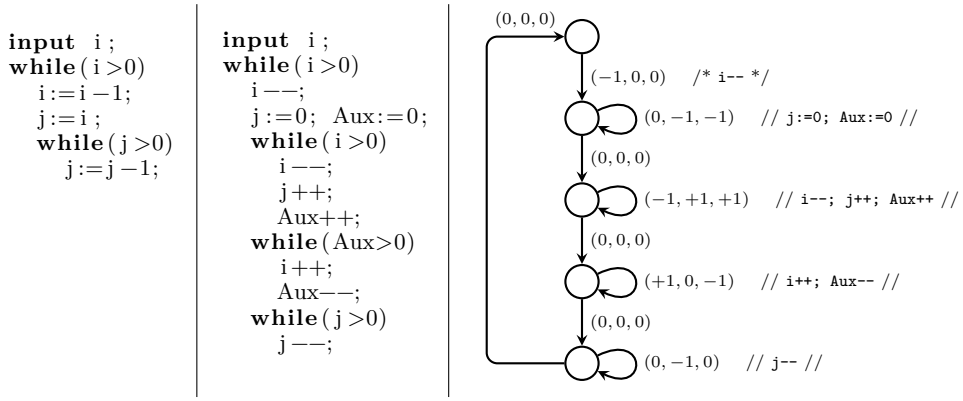

Figure 1 A skeleton of a simple imperative program (left) and its VASS model (right).

$\mathcal{C}[c] : \mathbb{N} \to \mathbb{N}_\infty$ such that $\mathcal{C}[c](n)$ is the maximal value of c along a computation initiated in a configuration with all counters set to n. So far, three types of VASS models have been investigated in previous works.

- *Demonic VASS*, where the non-determinism is resolved by an adversarial environment aiming to increase the complexity.
- *VASS Games*, where every control state is declared as *angelic* or *demonic*, and the non-determinism is resolved by the controller or by the environment aiming to lower and increase the complexity, respectively.
- *VASS MDPs*, where the states are either non-deterministic or stochastic. The non-determinism is usually resolved in the "demonic" way.

Let us note that the "angelic" and "demonic" non-determinism are standard concepts in program analysis [6] applicable to arbitrary computational devices including VASS. The use of VASS termination/counter complexity analysis is illustrated in the next example.

▶ **Example 1.** Consider the program skeleton of Fig. 1 (left). Since a VASS cannot directly model the assignment `j:=i` and cannot test a counter for zero, the skeleton is first transformed into an equivalent program of Fig. 1 (middle), where the assignment `j:=i` is implemented using an auxiliary variable `Aux` and two `while` loops. Clearly, the execution of the transformed program is only longer than the execution of the original skeleton (for all inputs). For the transformed program, an over-approximating demonic VASS model is obtained by replacing conditionals with non-determinism, see Fig. 1 (right). When all counters are initialized to n, the VASS terminates after $\mathcal{O}(n^2)$ transitions. Hence, the same upper bound is valid also for the original program skeleton. Actually, the run-time complexity of the skeleton is $\Theta(n^2)$ where n is the initial value of `i`, so the obtained upper bound is asymptotically optimal.

Existing results. In [5], it is shown that the problem whether $\mathcal{L} \in \mathcal{O}(n)$ for a given demonic VASS is solvable in polynomial time, and a complete proof method based on linear ranking functions is designed. The polynomiality of termination complexity for a given demonic VASS is also decidable in polynomial time, and if $\mathcal{L} \notin \mathcal{O}(n^k)$ for any $k \in \mathbb{N}$, then $\mathcal{L} \in 2^{\Omega(n)}$ [14]. The same results hold for counter complexity. In [20], a polynomial time algorithm computing the *least* $k \in \mathbb{N}$ such that $\mathcal{L} \in \mathcal{O}(n^k)$ for a given demonic VASS is presented (the algorithm first checks if such a k exists). It is also shown that if $\mathcal{L} \notin \mathcal{O}(n^k)$, then $\mathcal{L} \in \Omega(n^{k+1})$. Again, the same results hold also for counter complexity. The proof is actually given only for *strongly connected demonic VASS*, and it is conjectured that a generalization to unrestricted

demonic VASS can be obtained by extending the presented construction (see the Introduction of [20]). In [13], it was shown that the problem whether the termination/counter complexity of a given demonic VASS belongs to a given level of Grzegorczyk hierarchy is solvable in polynomial time, and the same problem for VASS games is shown **NP**-complete. The **NP** upper bound follows by observing that player Angel can safely commit to a *countreless*[1] strategy when minimizing the complexity level in the Grzegorczyk hierarchy. Intuitively, this is because Grzegorczyk classes are closed under function composition (unlike the classes $\Theta(n^k)$). Furthermore, the problem whether $\mathcal{L} \in \mathcal{O}(n^2)$ for a given VASS game is shown **PSPACE** hard, but the decidability of this problem is left open. As for VASS MDPs, the only existing result is [4], where it is shown that the linearity of termination complexity is solvable in polynomial time for VASS MDPs with a tree-like MEC decomposition.

Our contribution. For demonic VASS, we *refute* the conjecture of [20] and prove that for general (not necessarily strongly connected) demonic VASS, the problem whether

- $\mathcal{L} \in \mathcal{O}(n^k)$ is in **P** for $k = 1$, and **coNP**-complete for $k \geq 2$;
- $\mathcal{L} \in \Omega(n^k)$ is in **P** for $k \leq 2$, and **NP**-complete for $k \geq 3$;
- $\mathcal{L} \in \Theta(n^k)$ is in **P** for $k = 1$, **coNP**-complete for $k = 2$, and **DP**-complete for $k \geq 3$.

The same results are proven also for counter complexity.

Since the demonic VASS constructed in our proofs are relatively complicated, we write them in a simple imperative language with a precisely defined VASS semantics. This allows to present the overall proof idea clearly and justify technical correctness by following the control flow of the VASS program, examining possible side effects of the underlying "gadgets", and verifying that the Demon does not gain anything by deviating from the ideal execution scenario.

When proving the upper bounds, we show that every path in the DAG of strongly connected components can be associated with the (unique) vector describing the maximal simultaneous increase of the counters. Here, the counters pumpable to exponential (or even larger) values require special treatment. We show that this vector is computable in polynomial time. Hence, the complexity of a given counter c is $\Omega(n^k)$ iff there is a path in the DAG such that the associated maximal increase of c is $\Omega(n^k)$. Thus, we obtain the **NP** upper bound, and the other upper bounds follow similarly. The crucial parameter characterizing hard-to-analyze instances is the number of different paths from a root to a leaf in the DAG decomposition, and tractable subclasses of demonic VASS are obtained by bounding this parameter. We refer to Section 3 for more details.

Then, we turn our attention to VASS games, where the problem of polynomial termination/counter complexity analysis requires completely new ideas. In [13], it was observed that the information about the "asymptotic counter increase performed so far" must be taken into account by player Angel when minimizing the complexity level in the polynomial hierarchy, and counterless strategies are therefore insufficient. However, it is not clear what information is needed to make optimal decisions, and whether this information is finitely representable. We show that player Angel can safely commit to a so-called *locking* strategy. A strategy for player Angel is *locking* if whenever a new angelic state p is visited, one of its outgoing transition is chosen and "locked" so that when p is revisited, the same locked transition is used. The locked transition choice may depend on the computational history and the transitions locked in previously visited angelic states. Then, we define a *locking decomposition*

[1] A strategy is *counterless* if the decision depends just on the control state of the configuration currently visited.

```
1  input i;
2  j:=0; k:=0; z:=0;
3  if condition    // demonic choice //
4       then while (i>0) do j++; k:=k+i; i--; done
5       else j:=i*i; k:=i;
6            while (i>0) do j:=j+k; i--; done
7  choose:         // angelic choice //
8       while (j>0) do j--; z++ done
9  or: while (k>0) do k--; z++ done
```

Figure 2 A simple program with both demonic and angelic non-determinism.

of a given VASS that plays a role similar to the DAG decomposition for demonic VASS. Using the locking decomposition, the existence of a suitable locking strategy for player Angel is decided by an alternating polynomial time algorithm (and hence in polynomial space). Thus, we obtain the following: For every VASS game, we have that $\mathcal{L}$ is either in $\mathcal{O}(n^k)$ or in $\Omega(n^{k+1})$. Furthermore, the problem whether

- $\mathcal{L} \in \mathcal{O}(n^k)$ is **NP**-complete for k=1 and **PSPACE**-complete for $k \geq 2$;
- $\mathcal{L} \in \Omega(n^k)$ is in **P** for k=1, **coNP**-complete for k=2, and **PSPACE**-complete for $k \geq 3$;
- $\mathcal{L} \in \Theta(n^k)$ is **NP**-complete for k=1 and **PSPACE**-complete for $k \geq 2$.

The same results hold also for counter complexity. Similarly to demonic VASS, tractable subclasses of VASS games are obtained by bounding the number of different paths in the locking decomposition.

The VASS model constructed in Example 1 is purely demonic. The use of VASS *games* in program analysis/synthesis is illustrated in the next example.

▶ **Example 2.** Consider the program of Fig. 2. The `condition` at line 3 is resolved by the environment in a demonic way. The two branches of `if-then-else` execute a code modifying the variables `j` and `k`. After that, the controller can choose one of the two **while**-loops at lines 8, 9 with the aim of keeping the value of `z` small. The question is how the size of `z` grows with the size of input if the controller makes optimal decisions. A closer look reveals that when the variable `i` is assigned n at line 1, then

- the values of `j` and `k` are $\Theta(n)$ and $\Theta(n^2)$ when the `condition` is evaluated to *true*;
- the values of `j` and `k` are $\Theta(n^2)$ and $\Theta(n)$ when the `condition` is evaluated to *false*.

Hence, the controller can keep `z` in $\Theta(n)$ if an optimal decision is taken. Constructing a VASS game model for the program of Fig. 2 is straightforward (the required gadgets are given in Fig. 3). Using the results of this paper, the above analysis can be performed *fully automatically*.

2 Preliminaries

The sets of integers and non-negative integers are denoted by $\mathbb{Z}$ and $\mathbb{N}$, respectively, and we use $\mathbb{N}_\infty$ to denote $\mathbb{N} \cup \{\infty\}$. The vectors of $\mathbb{Z}^d$ where $d \geq 1$ are denoted by $\mathbf{v}, \mathbf{u}, \ldots$, and the vector $(n, \ldots, n)$ is denoted by $\vec{n}$.

▶ **Definition 3** (VASS). *Let $d \geq 1$. A* d-dimensional vector addition system with states (VASS) *is a pair $\mathcal{A} = (Q, Tran)$, where $Q \neq \emptyset$ is a finite set of* states *and $Tran \subseteq Q \times \mathbb{Z}^d \times Q$ is a finite set of* transitions *such that for every $q \in Q$ there exist $p \in Q$ and $\mathbf{u} \in \mathbb{Z}^d$ such that $(q, \mathbf{u}, p) \in Tran$.*

The set Q is split into two disjoint subsets Q_A and Q_D of *angelic* and *demonic* states controlled by the players Angel and Demon, respectively. A *configuration* of $\mathcal{A}$ is a pair $p\mathbf{v} \in Q \times \mathbb{N}^d$, where $\mathbf{v}$ is the vector of counter values. We often refer to counters by their symbolic names. For example, when we say that $\mathcal{A}$ has three counters x, y, z and the value of x in a configuration $p\mathbf{v}$ is 8, we mean that $d = 3$ and $\mathbf{v}_i = 8$ where i is the index associated to x. When the mapping between a counter name and its index is essential, we use c_i to denote the counter with index i.

A *finite path* in $\mathcal{A}$ of length m is a finite sequence $\varrho = p_1, \mathbf{u}_1, p_2, \mathbf{u}_2, \ldots, p_m$ such that $(p_i, \mathbf{u}_i, p_{i+1}) \in \mathit{Tran}$ for all $1 \leq i < m$. We use $\Delta(\varrho)$ to denote the *effect* of ϱ, defined as $\sum_{i=1}^{m} \mathbf{u}_i$. An *infinite path* in $\mathcal{A}$ is an infinite sequence $\alpha = p_1, \mathbf{u}_1, p_2, \mathbf{u}_2, \ldots$ such that every finite prefix $p_1, \mathbf{u}_1, \ldots, p_m$ of α is a finite path in $\mathcal{A}$.

A *computation* of $\mathcal{A}$ is a sequence of configurations $\alpha = p_1\mathbf{v}_1, p_2\mathbf{v}_2, \ldots$ of length $m \in \mathbb{N}_\infty$ such that for every $1 \leq i < m$ there is a transition $(p_i, \mathbf{u}_i, p_{i+1})$ satisfying $\mathbf{v}_{i+1} = \mathbf{v}_i + \mathbf{u}_i$. Note that every computation determines its associated path in the natural way.

VASS Termination Complexity. A *strategy* for Angel (or Demon) in $\mathcal{A}$ is a function η assigning to every finite computation $p_1\mathbf{v}_1, \ldots, p_m\mathbf{v}_m$ where $p_m \in Q_A$ (or $p_m \in Q_D$) a transition $(p_m, \mathbf{u}, q)$. Every pair of strategies (σ, π) for Angel/Demon and every initial configuration $p\mathbf{v}$ determine the unique *maximal* computation $\mathit{Comp}^{\sigma,\pi}(p\mathbf{v})$ initiated in $p\mathbf{v}$. The maximality means that the computation cannot be prolonged without making some counter negative. For a given counter c, we use $\mathit{max}[c](\mathit{Comp}^{\sigma,\pi}(p\mathbf{v}))$ to denote the supremum of the c's values in all configurations visited along $\mathit{Comp}^{\sigma,\pi}(p\mathbf{v})$. Furthermore, we use $\mathit{len}(\mathit{Comp}^{\sigma,\pi}(p\mathbf{v}))$ to denote the length of $\mathit{Comp}^{\sigma,\pi}(p\mathbf{v})$. Note that $\mathit{max}[c]$ and len can be infinite for certain computations.

For every initial configuration $p\mathbf{v}$, consider a game where the players Angel and Demon aim at minimizing and maximizing the $\mathit{max}[c]$ or len objective, respectively. By applying standard game-theoretic arguments (see [1] for an explicit proof), we obtain

$$\sup_{\pi} \inf_{\sigma} \mathit{len}(\mathit{Comp}^{\sigma,\pi}(p\mathbf{v})) = \inf_{\sigma} \sup_{\pi} \mathit{len}(\mathit{Comp}^{\sigma,\pi}(p\mathbf{v})) \tag{1}$$

$$\sup_{\pi} \inf_{\sigma} \mathit{max}[c](\mathit{Comp}^{\sigma,\pi}(p\mathbf{v})) = \inf_{\sigma} \sup_{\pi} \mathit{max}[c](\mathit{Comp}^{\sigma,\pi}(p\mathbf{v})) \tag{2}$$

where σ and π range over all strategies for Angel and Demon, respectively. Hence, there exists a unique *termination value* of $p\mathbf{v}$, denoted by $\mathit{Tval}(p\mathbf{v})$, defined by (1). Similarly, for every counter c there exists a unique *maximal counter value*, denoted by $\mathit{Cval}[c](p\mathbf{v})$, defined by (2). Furthermore, both players have *optimal* positional strategies σ^* and π^* achieving the outcome specified by the equilibrium value or better in every configuration $p\mathbf{v}$ against every strategy of the opponent (here, a *positional* strategy is a strategy depending only on the currently visited configuration). We refer to [1] for details.

The *termination complexity* and *c-counter complexity* of $\mathcal{A}$ are functions $\mathcal{L}, \mathcal{C}[c] : \mathbb{N} \to \mathbb{N}_\infty$ where $\mathcal{L}(n) = \max\{\mathit{Tval}(p\vec{n}) \mid p \in Q\}$ and $\mathcal{C}[c](n) = \max\{\mathit{Cval}[c](p\vec{n}) \mid p \in Q\}$. When the underlying VASS $\mathcal{A}$ is not clear, we write $\mathcal{L}_\mathcal{A}$ and $\mathcal{C}_\mathcal{A}[c]$ instead of $\mathcal{L}$ and $\mathcal{C}[c]$.

Observe that the asymptotic analysis of termination complexity for a given VASS $\mathcal{A}$ is trivially reducible to the asymptotic analysis of counter complexity in a VASS $\mathcal{B}$ obtained from $\mathcal{A}$ by adding a fresh "step counter" sc incremented by every transition of $\mathcal{B}$. Clearly, $\mathcal{L}_\mathcal{A} \in \Theta(\mathcal{C}_\mathcal{B}[sc])$. Therefore, the lower complexity bounds for the considered problems of asymptotic analysis are proven for $\mathcal{L}$, while the upper bounds are proven for $\mathcal{C}[c]$.

VASS Program 1 $\mathcal{A}_\varphi$.

```
1 d_2 += d_1 * e_1;  d_3 += d_2 * e_2;  ···  ;  d_k += d_{k-1} * e_{k-1};
2 foreach i = 1, ..., v do
3 |  choose:  x_i += d_k  or  x̄_i += d_k;
4 end
5 s_0 += d_k;
6 foreach i = 1, ..., m do
7 |  choose:  s_i += min(ℓ^i_1, s_{i-1})  or  s_i += min(ℓ^i_2, s_{i-1})  or  s_i += min(ℓ^i_3, s_{i-1});
8 end
9 f += s_m * n
```

3 Demonic VASS

In this section, we classify the computational complexity of polynomial asymptotic analysis for demonic VASS. The following theorem holds regardless whether the counter update vectors are encoded in unary or binary (the lower bounds hold for unary encoding, the upper bounds hold for binary encoding).

▶ **Theorem 4.** *Let $k \geq 1$. For every demonic VASS $\mathcal{A}$ we have that $\mathcal{L}$ is either in $\mathcal{O}(n^k)$ or in $\Omega(n^{k+1})$. Furthermore, the problem whether*

- *$\mathcal{L} \in \mathcal{O}(n^k)$ is in* **P** *for $k = 1$, and* **coNP***-complete for $k \geq 2$;*
- *$\mathcal{L} \in \Omega(n^k)$ is in* **P** *for $k \leq 2$, and* **NP***-complete for $k \geq 3$;*
- *$\mathcal{L} \in \Theta(n^k)$ is in* **P** *for $k = 1$,* **coNP***-complete for $k = 2$, and* **DP***-complete for $k \geq 3$.*

The same results hold also for $\mathcal{C}[c]$ (for a given counter c of $\mathcal{A}$).

The next theorem identifies the crucial parameter influencing the complexity of polynomial asymptotic analysis for demonic VASS. Let $\mathcal{D}(\mathcal{A})$ be the standard DAG of strongly connected components of $\mathcal{A}$. For every leaf (bottom SCC) η of $\mathcal{D}(\mathcal{A})$, let $Deg(\eta)$ be the total number of all paths from a root of $\mathcal{D}(\mathcal{A})$ to η.

▶ **Theorem 5.** *Let Λ be a class of demonic VASS such that for every $\mathcal{A} \in \Lambda$ and every leaf η of $\mathcal{D}(\mathcal{A})$ we have that $Deg(\eta)$ is bounded by a fixed constant depending only on Λ.*

Then, the problems whether $\mathcal{L}_\mathcal{A} \in \mathcal{O}(n^k)$, $\mathcal{L}_\mathcal{A} \in \Omega(n^k)$, $\mathcal{L}_\mathcal{A} \in \Theta(n^k)$ for given $\mathcal{A} \in \Lambda$ and $k \in \mathbb{N}$, are solvable in polynomial time (where the k is written in binary). The same results hold also for $\mathcal{C}[c]$ (for a given counter c of $\mathcal{A}$).

The degree of the polynomial bounding the running time of the decision algorithm for the three problems of Theorem 5 increases with the increasing size of the constant bounding $Deg(\eta)$. From the point of view of program analysis, Theorem 5 has a clear intuitive meaning. If $\mathcal{A}$ is an abstraction of a program $\mathcal{P}$, then the instructions in $\mathcal{P}$ increasing the complexity of the asymptotic analysis of $\mathcal{A}$ are *branching instructions* such as **if-then-else** that are *not embedded within loops*. If $\mathcal{P}$ executes many such constructs in a sequence, a termination point can be reached in many ways ("zigzags" in the $\mathcal{P}$'s control-flow graph). This increases $Deg(\eta)$, where η is a leaf of $\mathcal{D}(\mathcal{A})$ containing the control state modeling the termination point of $\mathcal{P}$.

3.1 Lower bounds

Since the asymptotic analysis of $\mathcal{L}$ is trivially reducible to the asymptotic analysis of $\mathcal{C}[c]$ (see Section 2), all lower complexity bounds of Theorem 4 follow directly from the next two lemmata.

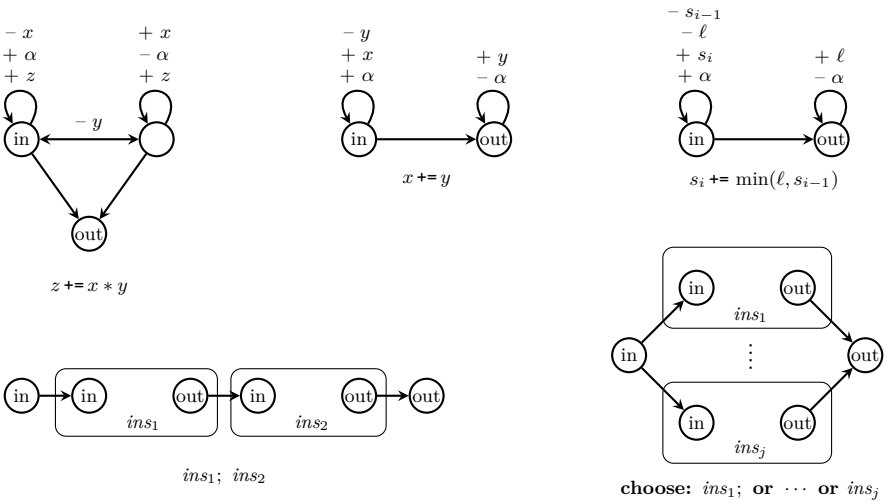

Figure 3 The gadgets of $\mathcal{A}_\varphi$.

▶ **Lemma 6.** *Let $k \geq 2$. For every propositional formula φ in 3-CNF there exists a demonic VASS $\mathcal{A}_\varphi$ constructible in time polynomial in $|\varphi|$ such that*

- *if φ is satisfiable, then $\mathcal{L}_{\mathcal{A}_\varphi} \in \Theta(n^{k+1})$;*
- *if φ is not satisfiable, then $\mathcal{L}_{\mathcal{A}_\varphi} \in \Theta(n^k)$.*

Proof. Let $\varphi \equiv C_1 \wedge \cdots \wedge C_m$ be a propositional formula where every $C_i \equiv \ell_1^i \vee \ell_2^i \vee \ell_3^i$ is a clause with three literals over propositional variables $X_1, \ldots, X_v$ (a literal is a propositional variable or its negation). We construct a VASS $\mathcal{A}_\varphi$ with the counters

- $x_1, \cdots, x_v, \bar{x}_1, \cdots, \bar{x}_v$ used to encode an assignment of truth values to $X_1, \ldots, X_v$. In the following, we identify literals ℓ_j^i of φ with their corresponding counters (i.e., if $\ell_j^i \equiv X_u$, the corresponding counter is x_u; and if $\ell_j^i \equiv \neg X_u$, the corresponding counter is $\bar{x}_u$).
- $s_0, \ldots, s_m$ used to encode the validity of clauses under the chosen assignment,
- f used to encode the (in)validity of φ under the chosen assignment,
- $d_1, \ldots, d_k$ and $e_1, \ldots, e_{k-1}$ used to compute n^k,
- and some auxiliary counters used in gadgets.

The structure of $\mathcal{A}_\varphi$ is shown in VASS Program 1. The basic instructions are implemented by the gadgets of Fig. 3 (top). Counter changes associated to a given transition are indicated by the corresponding labels, where $-c$ and $+c$ mean decrementing and incrementing a given counter by one (the other counters are unchanged). Hence, the empty label represents no counter change, i.e., the associated counter update vector is $\vec{0}$. The auxiliary counter α is *unique for every instance* of these gadgets and it is not modified anywhere else.

The constructs ins_1; ins_2 and **choose:** ins_1; **or** $\cdots$ **or** ins_j are implemented by connecting the underlying gadgets as shown in Fig. 3 (bottom). The **foreach** statements are just concise representations of the corresponding sequences of instructions connected by ';'.

Now suppose that the computation of VASS Program 1 is executed from line 1 where all counters are initialized to n. One can easily verify that all gadgets implement the operations associated to their labels up to some "asymptotically irrelevant side effects". More precisely,

- the z **+=** $x * y$ gadget ensures that the Demon can increase the value of counter z by $val(x) + val(y) \cdot (val(x) + n)$ (but not more) if he plays optimally, where $val(x)$ and $val(y)$ are the values stored in x and y when initiating the gadget. Recall that the counter α is unique for the gadget, and its initial value is n. Also note that the value of y is decreased to 0 when the Demon strives to maximally increase the value of z.

- The x += y gadget ensures that the Demon can add $val(y)$ to the counter x and then reset y to the value $val(y) + n$ (but not more) if he plays optimally. Again, note that α is a unique counter for the gadget with initial value n.
- The s_i += $\min(\ell, s_{i-1})$ gadget allows the Demon to increase s_i by the minimum of $val(\ell)$ and $val(s_{i-1})$, and then restore ℓ to $val(\ell) + n$ (but not more).

Now, the VASS Program 1 is easy to follow. We describe its execution under the assumption that the Demon plays *optimally*. It is easy to verify that the Demon cannot gain anything by deviating from the below described scenario where certain counters are pumped to their *maximal* values (in particular, the auxiliary counters are never re-used outside their gadgets, hence the Demon is not motivated to leave any positive values in them).

By executing line 1, the Demon pumps the counter d_k to the value $\Theta(n^k)$. Then, the Demon determines a truth assignment for every X_i, where $i \in \{1, \ldots, v\}$, by pumping either the counter x_i or the counter $\bar{x}_i$ to the value $\Theta(n^k)$. A key observation is that when the chosen assignment makes φ true, then every clause contains a literal such that the value of its associated counter is $\Theta(n^k)$. Otherwise, there is a clause C_i such that all of the three counters corresponding to ℓ_1^i, ℓ_2^i, ℓ_3^i have the value n. The Demon continues by pumping s_0 to the value $\Theta(n^k)$ at line 5. Then, for every $i = 1, \ldots, m$, he selects a literal ℓ_j^i of C_i and pumps s_i to the minimum of $val(s_{i-1})$ and $val(\ell_j^i)$. Observe that $val(s_{i-1})$ is either $\Theta(n)$ or $\Theta(n^k)$, and the same holds for $val(s_i)$ after executing the instruction. Hence, s_m is pumped either to $\Theta(n^k)$ or $\Theta(n)$, depending on whether the chosen assignment sets every clause to true or not, respectively. Note that the length of the whole computation up to line 9 is $\Theta(n^k)$, regardless whether the chosen assignment sets the formula φ to true or false. If s_m was pumped to $\Theta(n^k)$, then the last instruction at line 9 can pump the counter f to $\Theta(n^{k+1})$ in $\Theta(n^{k+1})$ transitions. Hence, if φ is satisfiable, the Demon can schedule a computation of length $\Theta(n^{k+1})$. Otherwise, the length of the longest computation is $\Theta(n^k)$. Also observe that if the Demon starts executing $\mathcal{A}_\varphi$ in some other control state (i.e., not in the first instruction of line 1), the maximal length of a computation is only shorter. ◀

Recall that the class **DP** consists of problems that are intersections of one problem in **NP** and another problem in **coNP**. The class **DP** is expected to be somewhat larger than $\mathbf{NP} \cup \mathbf{coNP}$, and it is contained in the $\mathbf{P^{NP}}$ level of the polynomial hierarchy. The standard **DP**-complete problem is SAT-UNSAT, where an instance is a pair φ, ψ of propositional formulae and the question is whether φ is satisfiable and ψ is unsatisfiable [17]. Hence, the **DP** lower bounds of Theorem 4 follow directly from the next lemma (a proof can be found in [1]).

▶ **Lemma 7.** *Let $k \geq 3$. For every pair φ, ψ of propositional formulae in 3-CNF there exists a demonic VASS $\mathcal{A}_{\varphi,\psi}$ such that $\mathcal{L}_{\mathcal{A}_{\varphi,\psi}} \in \Theta(n^k)$ iff φ is satisfiable and ψ is unsatisfiable.*

3.2 Upper bounds

The upper complexity bounds of Theorem 4 are proven for $\mathcal{C}[c]$. For the sake of clarity, we first sketch the main idea and then continue with developing a formal proof.

Intuition. For a given demonic VASS $\mathcal{A}$, we compute its SCC decomposition and proceed by analyzing the individual SCCs in the way indicated in Fig. 4. We start in a top SCC with all counters initialized to n. Here, we can directly apply the results of [20, 14] and decide in polynomial time whether $\mathcal{C}[c] \in \Theta(n^k)$ for some $k \in \mathbb{N}$ or $\mathcal{C}[c] \in 2^{\Omega(n)}$ (in the first case, we can also determine the k). We perform this analysis for every counter c and thus obtain the vector describing the maximal asymptotic growth of the counters (such as $(n^2, n, 2^{\Omega(n)})$ in Fig. 4). Observe that

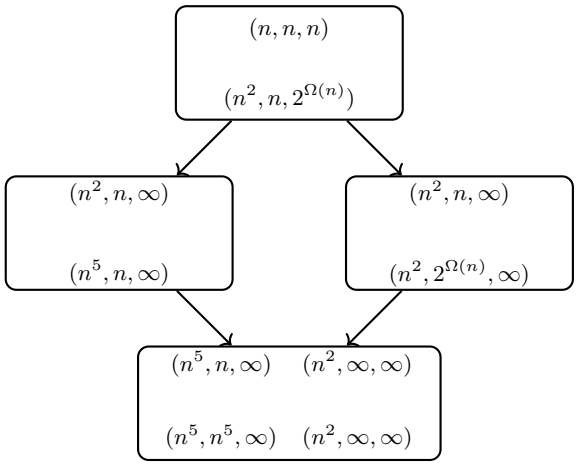

Figure 4 Analyzing $\mathcal{C}[c]$ in a demonic VASS by SCC decomposition.

- although the asymptotic growth of $\mathcal{C}[c]$ has been analyzed for each counter independently, all counters can be pumped to their associated asymptotic values *simultaneously*. Intuitively, this is achieved by considering the "pumping computations" for the smaller vector $(\lfloor n/d \rfloor, \ldots, \lfloor n/d \rfloor)$ of initial counter values (d is the dimension of $\mathcal{A}$), and then simply "concatenating" these computations in a configuration with all counters initialized to n;
- if the asymptotic growth of $\mathcal{C}[c]$ is $\Theta(n)$, the computation simultaneously pumping the counters to their asymptotic values may actually *decrease* the value of c (the computation can be arranged so that the resulting value of c stays above $\lfloor n/d \rfloor$ for all sufficiently large n). For example, the top SCC of Fig. 4 achieves the simultaneous asymptotic growth of all counters from (n, n, n) to $(n^2, n, 2^{\Omega(n)})$, but this does *not* imply the counters can be simultaneously increased above the original value n (nevertheless, the simultaneous increase in the first and the third counter above n is certainly possible for all sufficiently large n).

A natural idea how to proceed with next SCCs is to perform a similar analysis for larger vectors of initial counter values. Since we are interested just in the asymptotic growth of the counters, we can safely set the initial value of a counter previously pumped to $\Theta(n^k)$ to *precisely* n^k. However, it is not immediately clear how to treat the counters previously pumped to $2^{\Omega(n)}$. We show that the length of a computation "pumping" the counters to their new asymptotic values in the considered SCC $\mathcal{C}$ is at most exponential in n. Consequently, the "pumping computation" in $\mathcal{C}$ can be constructed so that the resulting value of the "large" counters stays above one half of their original value. This means the value of "large" counters is still in $2^{\Omega(n)}$ after completing the computation in $\mathcal{C}$. Furthermore, the large counters can be treated as if their initial value was infinite when analyzing $\mathcal{C}$. This "infinite" initial value is implemented simply by modifying every counter update vector $\mathbf{u}$ in $\mathcal{C}$ so that $\mathbf{u}[j] = 0$ for every "large" counter c_j. This adjustment in the structure of $\mathcal{C}$ is denoted by putting "∞" into the corresponding component of the initial counter value vector (see Fig. 4). This procedure is continued until processing all SCCs. Note that the same SCC may be processed *multiple times* for different vectors of initial counter values corresponding to different paths from a top SCC. In Fig. 4, the bottom SCC is processed for the initial vectors (n^5, n, ∞) and (n^2, ∞, ∞) corresponding to the two paths from the top SCC. The number of such initial vectors can be *exponential* in the size of $\mathcal{A}$, as witnessed by the VASS constructed in the proof of Lemma 6.

Now we give a formal proof. Let $\mathcal{A}$ be a demonic VASS with d counters. For every counter c and every $\mathbf{v} \in \mathbb{N}^d$, we define the function $\mathcal{C}[c, \mathbf{v}] : \mathbb{N} \to \mathbb{N}_\infty$ where $\mathcal{C}[c, \mathbf{v}](n)$ is the maximum of all $Cval[c](p\mathbf{u})$ where $p \in Q$ and $\mathbf{u} = (n^{\mathbf{v}(1)}, \ldots, n^{\mathbf{v}(d)})$.

▶ **Proposition 8.** *Let $\mathcal{A}$ be a strongly connected demonic VASS with d counters, and let $\mathbf{v} \in \mathbb{N}^d$ such that $\mathbf{v}(i) \leq 2^{j \cdot d}$ for every $i \leq d$, where $j < |Q|$. For every counter c, we have that either $\mathcal{C}[c, \mathbf{v}] \in \Theta(n^k)$ for some $1 \leq k \leq 2^{(j+1) \cdot d}$, or $\mathcal{C}[c, \mathbf{v}] \in \Omega(2^n)$. It is decidable in polynomial time which of the two possibilities holds. In the first case, the value of k is computable in polynomial time.*

In [20], a special variant of Proposition 8 covering the subcase when $\mathbf{v} = \vec{1}$ is proven. In the introduction part of [20], it is mentioned that a generalization of this result (equivalent to Proposition 8) can be obtained by modifying the techniques presented in [20]. Although no explicit proof is given, the modification appears feasible. We give a simple explicit proof of Proposition 8, using the algorithm of [20] for the $\mathbf{v} = \vec{1}$ subcase as a "black-box procedure". We refer to [1] for details.

Now we extend the function $\mathcal{C}[c, \mathbf{v}]$ so that $\mathbf{v} \in \mathbb{N}^d_\infty$. Here, the ∞ components of $\mathbf{v}$ correspond to counters that have already been pumped to "very large" values and do not constrain the computations in $\mathcal{A}$. As we shall see, "very large" actually means "at least singly exponential in n".

Let $\mathbf{v} \in \mathbb{N}^d_\infty$, and let $\mathcal{A}_\mathbf{v}$ be the VASS obtained from $\mathcal{A}$ by modifying every counter update vector $\mathbf{u}$ into $\mathbf{u}'$, where $\mathbf{u}'(i) = \mathbf{u}(i)$ if $\mathbf{v}(i) \neq \infty$, otherwise $\mathbf{u}'(i) = 0$. Hence, the counters set to ∞ in $\mathbf{v}$ are never changed in $\mathcal{A}_\mathbf{v}$. Furthermore, let $\mathbf{v}'$ be the vector obtained from $\mathbf{v}$ by changing all ∞ components into 1. We put $\mathcal{C}_\mathcal{A}[c, \mathbf{v}] = \mathcal{C}_{\mathcal{A}_\mathbf{v}}[c, \mathbf{v}']$.

For a given $\mathbf{v} \in \mathbb{N}^d_\infty$, we say that $F : \mathbb{N} \to \mathbb{N}^d$ is $\mathbf{v}$-*consistent* if for every $i \in \{1, \ldots, d\}$ we have that the projection $F_i : \mathbb{N} \to \mathbb{N}$ is either $\Theta(n^k)$ if $\mathbf{v}_i = k$, or $2^{\Omega(n)}$ if $\mathbf{v}_i = \infty$. Intuitively, a $\mathbf{v}$-consistent function assigns to every $n \in \mathbb{N}$ a vector $F(n)$ of initial counter values growing consistently with $\mathbf{v}$.

Given $\mathbf{v} \in \mathbb{N}^d_\infty$, a control state $p \in Q$, a $\mathbf{v}$-consistent function F, an infinite family $\Pi = \pi_1, \pi_2, \ldots$ of Demon's strategies in $\mathcal{A}$, a function $S : \mathbb{N} \to \mathbb{N}$, and $n \in \mathbb{N}$, we use $\beta_n[\mathbf{v}, p, F, \Pi, S]$ to denote the computation of $\mathcal{A}$ starting at $pF(n)$ obtained by applying π_n until a maximal computation is produced or $S(n)$ transitions are executed.

The next lemma says that if $\mathcal{A}$ is strongly connected, then all counters can be pumped *simultaneously* to the values asymptotically equivalent to $\mathcal{C}_\mathcal{A}[c, \mathbf{v}]$ so that the counters previously pumped to exponential values stay exponential.

▶ **Lemma 9.** *Let $\mathcal{A}$ be a strongly connected demonic VASS with d counters. Let $\mathbf{v} \in \mathbb{N}^d_\infty$, and let F be a $\mathbf{v}$-consistent function. Then for every counter c_i such that $\mathbf{v}_i \neq \infty$ and $\mathcal{C}_\mathcal{A}[c_i, \mathbf{v}] \in \Theta(n^k)$ we have that $Cval[c_i](pF(n)) \in \Theta(n^k)$ for every $p \in Q$. Furthermore, there exist $p \in Q$, an infinite family Π of Demon's strategies, and a function $S \in 2^{\mathcal{O}(n)}$ such that for every c_i, the value of c_i in the last configuration of $\beta_n[\mathbf{v}, p, F, \Pi, S]$ is*

- *$\Theta(n^k)$ if $\mathcal{C}_\mathcal{A}[c_i, \mathbf{v}] \in \Theta(n^k)$;*
- *$2^{\Omega(n)}$ if $\mathbf{v}_i = \infty$ or $\mathcal{C}_\mathcal{A}[c_i, \mathbf{v}] \in 2^{\Omega(n)}$.*

A proof of Lemma 9 uses the result of [14] saying that counters pumpable to exponential values can be simultaneously pumped by a computation of exponential length from a configuration where all counters are set to n (the same holds for polynomially bounded counters, where the length of the computation can be bounded even by a polynomial). Using the construction of Proposition 8, these results are extended to our setting with $\mathbf{v}$-consistent initial counter values. Then, the initial counter values are virtually "split into d boxes" of

size $\lfloor F(n)/d \rfloor$. The computations pumping the individual counters are then run "each in its own box" for these smaller initial vectors and concatenated. As the computation of one "box" cannot affect any other "box", no computation can undo the effects of previous computations. The details can be found in [1].

Let $\mathcal{V}_{\mathcal{A}} : \mathbb{N}_\infty^d \to \mathbb{N}_\infty^d$ be a function such that, for every $\mathbf{v} \in \mathbb{N}_\infty^d$,

$$\mathcal{V}_{\mathcal{A}}(\mathbf{v})(i) = \begin{cases} k & \text{if } \mathbf{v}_i \neq \infty \text{ and } \mathcal{C}_{\mathcal{A}}[c_i, \mathbf{v}] \in \Theta(n^k), \\ \infty & \text{otherwise.} \end{cases}$$

Note that every SCC (vertex) η of $\mathcal{D}(\mathcal{A})$ can be seen as a strongly connected demonic VASS after deleting all transitions leading from/to the states outside η. If the counters are simultaneously pumped to $\mathbf{v}$-consistent values before entering η, then η can further pump the counters to $\mathcal{V}_\eta(\mathbf{v})$-consistent values (see Lemma 9). According to Lemma 8, $\mathcal{V}_\eta(\mathbf{v})$ is computable in polynomial time for every $\mathbf{v} \in \mathbb{N}_\infty^d$ where every *finite* $\mathbf{v}_i$ is bounded by $2^{j \cdot d}$ for some $j < |Q|$.

Observe that *all* computations of $\mathcal{A}$ can be divided into finitely many pairwise disjoint classes according to their corresponding paths in $\mathcal{D}_{\mathcal{A}}$ (i.e., the sequence of visited SCCs of $\mathcal{D}_{\mathcal{A}}$). For each such sequence $\eta_1, \ldots, \eta_m$, the vectors $\mathbf{v}_0, \ldots, \mathbf{v}_m$ where $\mathbf{v}_0 = \vec{1}$ and $\mathbf{v}_i = \mathcal{V}_{\eta_i}(\mathbf{v}_{i-1})$ are computable in time polynomial in $|\mathcal{A}|$ (note that $m \leq |Q|$). The asymptotic growth of the counters achievable by computations following the path $\eta_1, \ldots, \eta_m$ is then given by $\mathbf{v}_m$. Hence, $\mathcal{C}_{\mathcal{A}}[c_i] \in \Omega(n^k)$ iff there is a path $\eta_1, \ldots, \eta_m$ in $\mathcal{D}_{\mathcal{A}}$ such that $\mathbf{v}_m(i) \geq k$. Similarly, $\mathcal{C}_{\mathcal{A}}[c_i] \in \mathcal{O}(n^k)$ iff for every path $\eta_1, \ldots, \eta_m$ in $\mathcal{D}_{\mathcal{A}}$ we have that $\mathbf{v}_m(i) \leq k$. From this we immediately obtain the upper complexity bounds of Theorem 4.

Furthermore, for every SCC η of $\mathcal{D}_{\mathcal{A}}$, we can compute the set $\mathit{Vectors}_{\mathcal{A}}(\eta)$ of *all* $\mathbf{u}$ such that there is a path $\eta_1, \ldots, \eta_m$ where η_1 is a root of $\mathcal{D}_{\mathcal{A}}$, $\eta_m = \eta$, and $\mathbf{u} = \mathbf{v}_m$. A full description of the algorithm is given in [1]. If $\mathit{Deg}(\eta)$ is bounded by a fixed constant independent of $\mathcal{A}$ for every leaf η of $\mathcal{D}_{\mathcal{A}}$, then the algorithm terminates in polynomial time, which proves Theorem 5.

4 VASS Games

The computational complexity of polynomial asymptotic analysis for VASS games is classified in our next theorem. The parameter characterizing hard instances is identified at the end of this section.

▶ **Theorem 10.** *Let $k \geq 1$. For every VASS game $\mathcal{A}$ we have that $\mathcal{L}$ is either in $\mathcal{O}(n^k)$ or in $\Omega(n^{k+1})$. Furthermore, the problem whether*

- *$\mathcal{L} \in \mathcal{O}(n^k)$ is* **NP***-complete for $k{=}1$ and* **PSPACE***-complete for $k{\geq}2$;*
- *$\mathcal{L} \in \Omega(n^k)$ is in* **P** *for $k{=}1$,* **coNP***-complete for $k{=}2$, and* **PSPACE***-complete for $k{\geq}3$;*
- *$\mathcal{L} \in \Theta(n^k)$ is* **NP***-complete for $k{=}1$ and* **PSPACE***-complete for $k{\geq}2$.*

The same results hold also for $\mathcal{C}[c]$ (for a given counter c of $\mathcal{A}$).

Furthermore, we show that for every VASS game $\mathcal{A}$, either $\mathcal{L} \in \mathcal{O}(n^{2^{d|Q|}})$ or $\mathcal{L} \in 2^{\Omega(n)}$. In the first case, the k such that $\mathcal{L} \in \Theta(n^k)$ can be computed in polynomial space. The same results hold for $\mathcal{C}[c]$.

In [13], it has been shown that the problem whether $\mathcal{L} \in \mathcal{O}(n)$ is **NP**-complete, and if $\mathcal{L} \notin \mathcal{O}(n)$, then $\mathcal{L} \in \Omega(n^2)$. This yields the **NP** and **coNP** bounds of Theorem 10 for $k = 1, 2$. Furthermore, it has been shown that the problem whether $\mathcal{L} \in \mathcal{O}(n^2)$ is **PSPACE**-hard, and this proof can be trivially generalized to obtain all **PSPACE** lower bounds of Theorem 10. The details are given in [1].

The key insight behind the proof of Theorem 10 is that player Angel can safely commit to a *simple locking* strategy when minimizing the counter complexity. We start by introducing locking strategies.

▶ **Definition 11.** *Let $\mathcal{A}$ be a VASS game. We say that a strategy σ for player Angel is* locking *if for every computation $p_1\mathbf{v}_1, \ldots, p_m\mathbf{v}_m$ where $p_m \in Q_A$ and for every $k < m$ such that $p_k = p_m$ we have that $\sigma(p_1\mathbf{v}_1, \ldots, p_k\mathbf{v}_k) = \sigma(p_1\mathbf{v}_1, \ldots, p_m\mathbf{v}_m)$.*

In other words, when an angelic control state p is visited for the first time, a locking strategy selects and "locks" an outgoing transition of p so that whenever p is revisited, the previously locked transition is taken. Observe that the choice of a "locked" transition may depend on the whole history of a computation.

Since a "locked" control state has only one outgoing transition, it can be seen as *demonic*. Hence, as more and more control states are locked along a computation, the VASS game $\mathcal{A}$ becomes "more and more demonic". We capture these changes as a finite acyclic graph $\mathcal{G}_\mathcal{A}$ called *the locking decomposition of $\mathcal{A}$*. Then, we say that a locking strategy is *simple* if the choice of a locked transition after performing a given history depends only on the finite path in $\mathcal{G}_\mathcal{A}$ associated to the history. We show that Angel can achieve an asymptotically optimal termination/counter complexity by using only simple locking strategies. Since the height of $\mathcal{G}_\mathcal{A}$ is polynomial in $|\mathcal{A}|$, the existence of an appropriate simple locking strategy for Angel can be decided by an alternating polynomial-time algorithm. As **AP** = **PSPACE**, this proves the **PSPACE** upper bounds of Theorem 10. Furthermore, our construction identifies the structural parameters of $\mathcal{G}_\mathcal{A}$ making the polynomial asymptotic analysis of VASS games hard. When these parameters are bounded by fixed constants, the problems of Theorem 10 are solvable in polynomial time.

4.1 Locking sets and the locking decomposition of $\mathcal{A}$

Let $\mathcal{A}$ be a VASS game. A *Demonic decomposion* of $\mathcal{A}$ is a finite directed graph $\mathcal{D}_\mathcal{A}$ defined as follows. Let $\sim \subseteq Q \times Q$ be an equivalence where $p \sim q$ iff either $p = q$, or both p, q are demonic and mutually reachable from each other via a finite path leading only through demonic control states. The vertices of $\mathcal{D}_\mathcal{A}$ are the equivalence classes $Q/\!\sim$, and $[p] \to [q]$ iff $[p] \neq [q]$ and $(p, \mathbf{u}, q) \in \mathit{Tran}$ for some $\mathbf{u}$. For demonic VASS, $\mathcal{D}_\mathcal{A}$ becomes the standard DAG decomposition. For VASS games, $\mathcal{D}_\mathcal{A}$ is *not* necessarily acyclic.

A *locking set* of $\mathcal{A}$ is a set of transitions $L \subseteq \mathit{Tran}$ such that $(p, \mathbf{u}, q) \in L$ implies $p \in Q_A$, and $(p, \mathbf{u}, q), (p', \mathbf{u}', q') \in L$ implies $p \neq p'$. A control state p is *locked* by L if L contains an outgoing transition of p. We use $\mathscr{L}$ to denote the set of all locking sets of $\mathcal{A}$. For every $L \in \mathscr{L}$, let $\mathcal{A}_L$ be the VASS game obtained from $\mathcal{A}$ by "locking" the transitions of L. That is, each control state p locked by L becomes demonic in $\mathcal{A}_L$, and the only outgoing transition of p in $\mathcal{A}_L$ is the transition $(p, \mathbf{u}, q) \in L$.

▶ **Definition 12.** *The* locking decomposition of $\mathcal{A}$ *is a finite directed graph $\mathcal{G}_\mathcal{A}$ where the set of vertices and the set of edges of $\mathcal{G}_\mathcal{A}$ are the least sets V and $\to$ satisfying the following conditions:*

- *All elements of V are pairs $([p], L)$ where $L \in \mathscr{L}$ and $[p]$ is a vertex of $\mathcal{D}_{\mathcal{A}_L}$. When p is demonic/angelic in $\mathcal{A}_L$, we say that $([p], L)$ is demonic/angelic.*
- *V contains all pairs of the form $([p], \emptyset)$.*
- *If $([p], L) \in V$ where p is demonic in $\mathcal{A}_L$ and $[p] \to [q]$ is an edge of $\mathcal{D}_{\mathcal{A}_L}$, then $([q], L) \in V$ and $([p], L) \to ([q], L)$.*
- *If $([p], L) \in V$ where p is angelic in $\mathcal{A}_L$, then for every $(p, \mathbf{u}, q) \in \mathit{Tran}$ we have that $([q], L') \in V$ and $([p], L) \to ([q], L')$, where $L' = L \cup \{(p, \mathbf{u}, q)\}$.*

It is easy to see that $\mathcal{G}_\mathcal{A}$ is *acyclic* and the length of every path in $\mathcal{G}_\mathcal{A}$ is bounded by $|Q| + |Q_A|$, where at most $|Q|$ vertices in the path are demonic. Note that every computation of $\mathcal{A}$ obtained by applying a locking strategy determines its associated path in $\mathcal{G}_\mathcal{A}$ in the natural way. A locking strategy σ is *simple* if the choice of a locked transition depends only on the path in $\mathcal{G}_\mathcal{A}$ associated to the performed history. (i.e., if two histories determine the same path in $\mathcal{G}_\mathcal{A}$, then the strategy makes the same choice for both histories).

4.2 Upper bounds

Let $\mathcal{A}$ be a VASS game with d counters. For every $p \in Q$ and $\mathbf{v} \in \mathbb{N}^d$, let $\mathcal{C}^p_\mathcal{A}[c, \mathbf{v}](n) = \mathit{Cval}[c](p\mathbf{u})$ where $\mathbf{u} = (n^{\mathbf{v}(1)}, \ldots, n^{\mathbf{v}(d)})$. We extend this notation to the vectors $\mathbf{v} \in \mathbb{N}^d_\infty$ in the same way as in Section 3.2, i.e., for a given $\mathbf{v} \in \mathbb{N}^d_\infty$, we put $\mathcal{C}^p_\mathcal{A}[c, \mathbf{v}] = \mathcal{C}^p_{\mathcal{A}_\mathbf{v}}[c, \mathbf{v}']$. Recall that $\mathbf{v}'$ is the vector obtained from $\mathbf{v}$ by changing all ∞ components into 1, and $\mathcal{A}_\mathbf{v}$ is the VASS obtained from $\mathcal{A}$ by modifying every counter update vector $\mathbf{u}$ into $\mathbf{u}'$, where $\mathbf{u}'(i) = \mathbf{u}(i)$ if $\mathbf{v}(i) \neq \infty$, otherwise $\mathbf{u}'(i) = 0$. The main technical step towards obtaining the **PSPACE** upper bounds of Theorem 10 is the next proposition.

▶ **Proposition 13.** *Let $\mathcal{A}$ be a VASS game with d counters. Furthermore, let $([p], L)$ be a vertex of $\mathcal{G}_\mathcal{A}$, $\mathbf{v} \in \mathbb{N}^d_\infty$, and c_i a counter such that $\mathbf{v}_i \neq \infty$. Then, one of the following two possibilities holds:*

- *there is $k \in \mathbb{N}$ such that for every $\mathbf{v}$-consistent F there exist a simple locking Angel's strategy $\sigma_\mathbf{v}$ in $\mathcal{A}_L$ and a Demon's strategy $\pi_\mathbf{v}$ in $\mathcal{A}_L$ such that $\sigma_\mathbf{v}$ is independent of F and*
  - *for every Demon's strategy π in $\mathcal{A}_L$, we have that $\mathit{max}[c_i](\mathit{Comp}^{\sigma_\mathbf{v},\pi}_{\mathcal{A}_L}(p\,F(n))) \in \mathcal{O}(n^k)$;*
  - *for every Angel's strategy σ in $\mathcal{A}_L$, we have that $\mathit{max}[c_i](\mathit{Comp}^{\sigma,\pi_\mathbf{v}}_{\mathcal{A}_L}(p\,F(n))) \in \Omega(n^k)$.*
- *for every $\mathbf{v}$-consistent F there is a Demon's strategy $\pi_\mathbf{v}$ in $\mathcal{A}_L$ such that for every Angel's strategy σ in $\mathcal{A}_L$, we have that $\mathit{max}[c_i](\mathit{Comp}^{\sigma,\pi_\mathbf{v}}_{\mathcal{A}_L}(p\,F(n))) \in 2^{\Omega(n)}$.*

Proposition 13 is proven by induction on the height of the subgraph rooted by $([p], L)$. The case when $([p], L)$ is demonic (which includes the base case when $([p], L)$ is a leaf) follows from the constructions used in the proof of Proposition 8. When the vertex $([p], L)$ is angelic, it has immediate successors of the form $([q_i], L_i)$ where $L_i = L \cup \{(p, \mathbf{u}_i, q_i)\}$. We show that by locking one of the $(p, \mathbf{u}_i, q_i)$ transitions in p, Angel can minimize the growth of c_i in asymptotically the same way as if he used all of these transitions freely when revisiting p. We refer to [1] for details.

Observe that every computation in $\mathcal{A}$ where Angel uses some simple locking strategy determines the unique corresponding path in $\mathcal{G}_\mathcal{A}$ (initiated in a vertex of the form $([p], \emptyset)$) in the natural way. Hence, all such computations can be divided into finitely many pairwise disjoint classes according to their corresponding paths in $\mathcal{G}_\mathcal{A}$. Let $([p_1], L_1), \ldots, ([p_k], L_k)$ be a path in $\mathcal{G}_\mathcal{A}$ where $L_1 = \emptyset$. Consider the corresponding sequence $\mathbf{v}_0, \ldots, \mathbf{v}_k$ where $\mathbf{v}_0 = \vec{1}$ and $\mathbf{v}_i$ is equal either to $\mathcal{V}_{[p_i]}(\mathbf{v}_{i-1})$ or to $\mathbf{v}_{i-1}$, depending on whether $([p_i], L_i)$ is demonic or angelic, respectively. Here, $\mathcal{V}$ is the function defined in Section 3.2 (observe that the component $[p]$ of $\mathcal{D}_{A_L}$ containing p can be seen as a strongly connected demonic VASS after deleting all transitions from/to the states outside $[p]$). The vector $\mathbf{v}_k$ describes the maximal asymptotic growth of the counters achievable by Demon when Angel uses the simple locking strategy associated to the path. Furthermore, the sequence $\mathbf{v}_0, \ldots, \mathbf{v}_k$ is computable in time polynomial in $|\mathcal{A}|$ and all finite components of $\mathbf{v}_k$ are bounded by $2^{d \cdot |Q|}$ because the total number of all demonic $([p_i], L_i)$ in the path is bounded by $|Q|$ (cf. Proposition 8).

The problem whether $\mathcal{C}[c_i] \in \mathcal{O}(n^k)$ can be decided by an *alternating* polynomial-time algorithm which selects an initial vertex of the form $([p], \emptyset)$ universally, and then constructs a maximal path in $\mathcal{G}_\mathcal{A}$ from $([p], \emptyset)$ where the successors of demonic/angelic

vertices are chosen universally/existentially, respectively. After obtaining a maximal path $([p_1], L_1), \ldots, ([p_k], L_k)$, the vector $\mathbf{v}_k$ is computed in polynomial time, and the algorithm answers yes/no depending on whether $\mathbf{v}_k(i) \leq k$ or not, respectively. The problem whether $\mathcal{C}[c_i] \in \Omega(n^k)$ is decided similarly, but here the initial vertex is chosen existentially, the successors of demonic/angelic vertices are chosen existentially/universally, and the algorithm answers yes/no depending on whether $\mathbf{v}_k(i) \geq k$ or not, respectively. This proves the **PSPACE** upper bounds of Theorem 10.

Observe that the crucial parameter influencing the computational hardness of the asymptotic analysis for VASS games is the number of maximal paths in $\mathcal{G}_\mathcal{A}$. If $|Q_A|$ and $Deg([p], L)$ are bounded by constants, then the above alternating polynomial time algorithms can be simulated by *deterministic* polynomial time algorithms. Thus, we obtain the following:

▶ **Theorem 14.** *Let Λ be a class of VASS games such that for every $\mathcal{A} \in \Lambda$ we have that $|Q_A|$ and $Deg([p], L)$, where $([p], L)$ is a leaf of $\mathcal{G}_\mathcal{A}$, are bounded by a fixed constant depending only on Λ. Then, the problems whether $\mathcal{L}_\mathcal{A} \in \mathcal{O}(n^k)$, $\mathcal{L}_\mathcal{A} \in \Omega(n^k)$, $\mathcal{L}_\mathcal{A} \in \Theta(n^k)$ for given $\mathcal{A} \in \Lambda$ and $k \in \mathbb{N}$, are solvable in polynomial time (where the k is written in binary). The same results hold also for $\mathcal{C}[c]$ (for a given counter c of $\mathcal{A}$).*

5 Conclusions, future work

We presented a precise complexity classification for the problems of polynomial asymptotic complexity of demonic VASS and VASS games. We also identified the structural parameters making these problems computationally hard, and we indicated that these parameters may actually stay reasonably small when dealing with VASS abstractions of computer programs. The actual applicability and scalability of the presented results to the problems of program analysis requires a more detailed study including experimental evaluation.

From a theoretical point of view, a natural question is whether the scope of effective asymptotic analysis can be extended from purely non-deterministic VASS to VASS with probabilistic transitions (i.e., VASS MDPs and VASS stochastic games). These problems are challenging and motivated by their applicability to probabilistic program analysis.

References

1 M. Ajdarów and A. Kučera. Deciding polynomial termination complexity for VASS programs. *arXiv*, 2102.06889 [cs.LO], 2021. `arXiv:2102.06889`.

2 E. Albert, P. Arenas, S. Genaim, M. Gómez-Zamalloa, G. Puebla, D. V. Ramírez-Deantes, G. Román-Díez, and D. Zanardini. Termination and cost analysis with COSTA and its user interfaces. *ENTCS*, 258(1):109–121, 2009. `doi:10.1016/j.entcs.2009.12.008`.

3 C. Alias, A. Darte, P. Feautrier, and L. Gonnord. Multi-dimensional rankings, program termination, and complexity bounds of flowchart programs. In *Proceedings of SAS 2010*, volume 6337 of *Lecture Notes in Computer Science*, pages 117–133. Springer, 2010. `doi:10.1007/978-3-642-15769-1_8`.

4 T. Brázdil, K. Chatterjee, A. Kučera, P. Novotný, and D. Velan. Deciding fast termination for probabilistic VASS with nondeterminism. In *Automated Technology for Verification and Analysis - 17th International Symposium, ATVA 2019, Taipei, Taiwan, October 28-31, 2019, Proceedings*, volume 11781 of *Lecture Notes in Computer Science*, pages 462–478. Springer, 2019. `doi:10.1007/978-3-030-31784-3_27`.

5 T. Brázdil, K. Chatterjee, A. Kučera, P. Novotný, D. Velan, and F. Zuleger. Efficient algorithms for asymptotic bounds on termination time in VASS. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, pages 185–194. ACM, 2018. `doi:10.1145/3209108.3209191`.

6 M. Broy and M. Wirsing. On the algebraic specification of nondeterministic programming languages. In *CAAP '81, Trees in Algebra and Programming, 6th Colloquium, Genoa, Italy, March 5-7, 1981, Proceedings*, volume 112 of *Lecture Notes in Computer Science*, pages 162–179. Springer, 1981. doi:10.1007/3-540-10828-9_61.

7 Q. Carbonneaux, J. Hoffmann, T. W. Reps, and Z. Shao. Automated resource analysis with coq proof objects. In *Proceedings of CAV 2017*, volume 10427 of *Lecture Notes in Computer Science*, pages 64–85. Springer, 2017. doi:10.1007/978-3-319-63390-9_4.

8 W. Czerwiński, S. Lasota, R. Lazić, J. Leroux, and F. Mazowiecki. The reachability problem for Petri nets is not elementary. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 24–33. ACM, 2019. doi:10.1145/3313276.3316369.

9 A. F.-Montoya and R. Hähnle. Resource analysis of complex programs with cost equations. In *Proceedings of APLAS 2014*, volume 8858 of *Lecture Notes in Computer Science*, pages 275–295. Springer, 2014. doi:10.1007/978-3-319-12736-1_15.

10 J. Giesl, C. Aschermann, M. Brockschmidt, F. Emmes, F. Frohn, C. Fuhs, J. Hensel, C. Otto, M. Plücker, P. Schneider-Kamp, T. Ströder, S. Swiderski, and R. Thiemann. Analyzing program termination and complexity automatically with aprove. *J. Autom. Reasoning*, 58(1):3–31, 2017. doi:10.1007/s10817-016-9388-y.

11 S. Gulwani, K. K. Mehra, and T. M. Chilimbi. SPEED: precise and efficient static estimation of program computational complexity. In *Proceedings of POPL 2009*, pages 127–139. ACM, 2009. doi:10.1145/1480881.1480898.

12 J. Hoffmann, K. Aehlig, and M. Hofmann. Multivariate amortized resource analysis. *ACM Trans. Program. Lang. Syst.*, 34(3):14:1–14:62, 2012. doi:10.1145/2362389.2362393.

13 A. Kučera, J. Leroux, and D. Velan. Efficient analysis of VASS termination complexity. In *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 676–688, 2020. doi:10.1145/3373718.3394751.

14 J. Leroux. Polynomial vector addition systems with states. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 134:1–134:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.134.

15 R. Lipton. The reachability problem requires exponential space. Technical report 62, Yale, 1976.

16 E.W. Mayr and A.R. Meyer. The complexity of the finite containment problem for Petri nets. *J. ACM*, 28(3):561–576, 1981. doi:10.1145/322261.322271.

17 Ch. Papadimitriou. *Computational Complexity*. Addison, 1994.

18 M. Sinn, F. Zuleger, and H. Veith. A simple and scalable static analysis for bound analysis and amortized complexity analysis. In *Proccedings of CAV 2014*, volume 8559 of *Lecture Notes in Computer Science*, pages 745–761. Springer, 2014. doi:10.1007/978-3-319-08867-9_50.

19 M. Sinn, F. Zuleger, and H. Veith. Complexity and resource bound analysis of imperative programs using difference constraints. *J. Autom. Reasoning*, 59(1):3–45, 2017. doi:10.1007/s10817-016-9402-4.

20 F. Zuleger. The polynomial complexity of vector addition systems with states. In *Foundations of Software Science and Computation Structures - 23rd International Conference, FOSSACS 2020, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2020, Dublin, Ireland, April 25-30, 2020, Proceedings*, volume 12077 of *Lecture Notes in Computer Science*, pages 622–641. Springer, 2020. doi:10.1007/978-3-030-45231-5_32.