Involved VASS Zoo

Wojciech Czerwiński
University of Warsaw, Poland

Abstract

We briefly describe recent advances on understanding the complexity of the reachability problem for vector addition systems (or equivalently for vector addition systems with states - VASSes). We present a zoo of a few involved VASS examples, which illustrate various aspects of hardness of VASSes and various techniques of proving lower complexity bounds.

2012 ACM Subject Classification Theory of computation → Parallel computing models

Keywords and phrases vector addition systems, reachability problem, low dimensions, examples

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2022.5

Category Invited Talk

Funding *Wojciech Czerwiński*: Supported by the ERC grant INFSYS, agreement no. 950398.

1 Introduction

Vector addition systems and essentially equivalent Petri nets are one of the most natural models of computation. They are also widely used in practise [21]. A convenient way to work with vector addition systems is to consider its extension by states (which is also essentially equivalent), namely vector addition systems with states (VASSes). A d-dimensional VASS (shortly a d-VASS) is a finite automaton equipped with d integer counters. Each transition can increase or decrease the counters by fixed values. Importantly, no counter can be ever decreased below zero. The counter represents the current number of items of some resource in the modelled system, thus it is natural to assume that this number is nonnegative.

▶ **Example 1.** The following 3-VASS was introduced in [9], we call it the *HP-gadget* after the names of authors of [9]. This VASS has interesting properties, which we use in the sequel. Transition colours are just to distinguish particular transitions, they have no semantics in the VASS behaviour.

$$\begin{tikzcd} p \arrow[loop left, "{(-1,1,0)}"] \arrow[r, bend left, "{(0,0,0)}"] & q \arrow[loop right, "{(2,-1,0)}"] \arrow[l, bend left, "{(0,0,-1)}"] \end{tikzcd}$$

The following is an example of a run

$$p(2,0,7) \longrightarrow p(1,1,7) \longrightarrow p(0,2,7) \longrightarrow q(0,2,7) \longrightarrow q(2,1,7) \longrightarrow q(4,0,7) \longrightarrow p(4,0,6)$$

Observe that in a similar way there is a run from $p(k,0,n)$ to $p(2k,0,n-1)$: we apply k times the blue transition reaching $p(0,k,n)$, then once the black transition reaching $q(0,k,n)$, then k times the green transition reaching $q(2k,0,n)$ and finally once the red transition reaching $p(2k,0,n-1)$. Intuitively in the state p we transfer value k from the first counter to the second one and then jump to state q. In the state q we transfer back value k to the first counter while multiplying it by 2. Finally we jump back to the state p decreasing the third counter by one. We will use similar approach many times in the sequel. Notice that repeating this process n times we have the following run

$$p(1,0,n) \longrightarrow p(2,0,n-1) \longrightarrow \ldots \longrightarrow p(2^{n-1},0,1) \longrightarrow p(2^n,0,0),$$

where each black arrow represents a sequence of transitions (in the sequel we often draw a sequence of transitions as one arrow). We also have

$$p(2^n,0,0) \longrightarrow p(x,y,0)$$

for any $x + y = 2^n$, so the set of configurations reachable from $p(1,0,n)$ is of at least exponential size.

On the other hand the size of the *reachability set* of $p(1,0,n)$ (set of configurations reachable from $p(1,0,n)$) is finite. Indeed, the red transition can be fired at most n times and it is easy to see that in between of two firings of the red transition all the other transitions also have to be fired only finitely many times. Thus the above example is the first interesting one: the reachability set is finite, but of at least exponential size (in that case of exactly exponential size).

Various decision problems for VASSes are studies since the 70-ties (with the proviso that in those times they were known under the name of Petri nets). Probably the most central one is the *reachability problem*. It asks whether in a given VASS there is a run from a given source *configuration*, to a given target configuration. A configuration is a state together with a counter valuation. Another related fundamental problem is the *coverability problem*, which asks whether in a given VASS there is a run from a given source configuration to a configuration which is *above* a given target configuration. We say that one configuration is above the other one if it has the same state, but counter values may be higher.

2 History of the problem

The reachability and coverability problems are considered since the 70-ties. The first milestone result was ExpSpace-hardness of the coverability problem by Lipton in 1976 [17]. Notice that this implies ExpSpace-hardness of the reachability problem, as coverability can be reduced to reachability by adding to a VASS additional transitions decreasing counters in the target state (one transition for each counter). In 1978 Rackoff has proven that the coverability problem is in ExpSpace [19]. He achieved it by showing that if there is a run from the source configuration s to some configuration $t' \succeq t$ (namely t' is above t) then there is also some *short* run from the source configuration s to some configuration $t'' \succeq t$, where by short be mean at most doubly-exponential in the input size. This approach, by small witness (which is often a short run) turns out to be successful in many cases for the reachability problem in VASSes. In 1982 finally decidability of the reachability problem was proved by Mayr [18]. The construction was very involved, so the follow-up works by Kosaraju and Lambert tried to simplify the solution and phrase it in a bit simpler setting [10, 11]. This construction is currently often known by the name KLM decomposition, as it decomposes the input VASS into many simpler ones.

After these breakthrough results there was a long period of not much progress on the reachability problem. The community tried to improve the state of art, but it was hard, so results about VASSes are scarce in the 90-ties. In 2009 Haase at al. proved that in 1-VASSes with numbers on transitions encoded in binary (we call such VASSes binary) the reachability problem is NP-complete [8]. It is easy to show that for unary 1-VASSes the problem is NL-complete. More progress on low dimensional VASSes followed. In 2015 Blondin at el. proved that in binary 2-VASSes the reachability problem is PSpace-complete [1], while a year later this result was improved by Englert at el. to NL-completeness in unary 2-VASSes [6]. Both the upper complexity bounds in dimension two were shown by the use of short run approach: authors of [1] proved that if there is any run from the source to the target in binary 2-VASS then there is also one of at most exponential length, while in [6] the same was shown for unary 2-VASSes and polynomial length runs.

Recently there was also a big progress in fixing complexity of the reachability problem. In 2015 Leroux and Schmitz have obtained first complexity upper bound on the problem [15]. By careful analysis of the KLM decomposition algorithm they proved that it runs in cubic-Ackermann time. In 2019 the same authors improved their previous result. They proposed a slight modification of the KLM decomposition algorithm and elegantly analysing the dimension a some vector spaces proved that the modified version runs in Ackermann time [16]. Also in 2019 Czerwiński et al. proved that the reachability problem is Tower-hard [2]. This was a surprise as many people felt that the problem should rather be ExpSpace-complete, but we probably lack some insight to prove the upper bound. In [2] we have used the technique of multiplication triples described later. Just two years later the complexity of the problem was finally settled to be Ackermann-complete. Two teams have independently shown Ackermann-hardness using slightly different techniques: Leroux [13] and Czerwiński and Orlikowski [4]. In [4] we have used the technique of controlling-counter and amplifiers, the technique of controlling-counter is described later.

3 Remaining challenges

Despite the fact that the complexity of the reachability problem is VASSes was established the problem still remains elusive in my opinion. The gap in our understanding is most striking in dimension three. For binary 2-VASSes the problem is PSpace-complete [1]. However for binary 3-VASSes the best complexity lower bound is still PSpace-complete inherited from the dimension two, while the best known upper bound is higher than Tower, namely in the $\mathcal{F}_7$ complexity class of the fast growing hierarchy [16]. We define the hierarchy of fast growing functions as $F_1(n) = 2n$ and $F_{k+1}(n) = \underbrace{F_{k-1} \circ \ldots \circ F_{k-1}}_{n}(1)$ for any $k > 1$. One can easily see that in particular $F_2(n) = 2^n$ and $F_3(n) = \text{Tower}(n)$. Based on the hierarchy of fast growing functions F_i one defines a hierarchy of fast growing complexity classes $\mathcal{F}_i$, which roughly speaking is the class of problems solvable in time F_i closed under a few natural operations [20]. Thus in particular we do not know whether existence of a run from the source to the target always implies existence of exponential length run or not. Or maybe length of this short run is doubly-exponential or tower size. Similarly we lack knowledge about other low dimensions.

Generally in dimension d the best upper bound for the reachability problem is $\mathcal{F}_{d+4}$ [16] (that is how we get $\mathcal{F}_7$ in dimension three). The current best lower complexity bound is $\mathcal{F}_d$-hardness in dimension $2d + 4$, so $\mathcal{F}_{(d-4)/2}$-hardness for d-VASSes [14]. The current research goal here is to find out whether we can get $\mathcal{F}_d$-hardness in dimension $d + C$ for some constant $C \in \mathbb{N}$.

Recently we worked with co-authors on the reachability problem for low dimensional VASSes [3, 5] motivated by the following two main ideas: 1) low dimensional VASSes are by itself a natural computation model, 2) understanding problems in low dimensional VASSes often turns out to be the best way of developing techniques very useful in general dimension. Indeed, understanding low dimensions was actually the triggering point for our results [2] and [4]. Current best complexity lower bounds for low dimensional VASSes are proven in our work with Łukasz Orlikowski [4]:

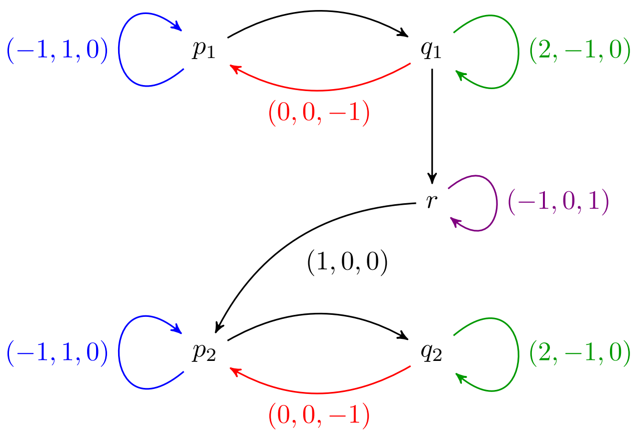
- NP-hardness for unary 4-VASSes;
- PSpace-hardness for unary 5-VASSes;
- ExpSpace-hardness for binary 6-VASSes;
- Tower-hardness for unary 8-VASSes.

The rest of this text focuses on presenting techniques of proving lower complexity bounds from the perspective of concrete low dimensions or concrete examples of involved low dimensional VASSes. We believe this perspective is the best way to illustrate the intuitions behind various approaches and to introduce various techniques useful in general.

4 Big finite reachability sets

We start the involved examples zoo from a family of examples, which is a folklore since years. These are VASSes, which have finite reachability set, but this set is very big. We first present a 3-VASS with finite, but doubly-exponential reachability set. For simplicity we do not write a vector on the transition if it does not change the counters at all (we often colour such transitions black).

▶ **Example 2.** The following 3-VASS has doubly-exponential reachability set.


$$\begin{tikzcd}
p_1 \arrow[loop left, "{(-1,1,0)}"] \arrow[rr, bend left] & & q_1 \arrow[loop right, "{(2,-1,0)}"] \arrow[ll, bend left, "{(0,0,-1)}"] \arrow[d] \\
& & r \arrow[loop right, "{(-1,0,1)}"] \arrow[dll, bend right, "{(1,0,0)}"] \\
p_2 \arrow[loop left, "{(-1,1,0)}"] \arrow[rr, bend left] & & q_2 \arrow[loop right, "{(2,-1,0)}"] \arrow[ll, bend left, "{(0,0,-1)}"]
\end{tikzcd}$$

Notice that the above example consists of two copies of the HP-gadget from Example 1. Thus we have the following run:

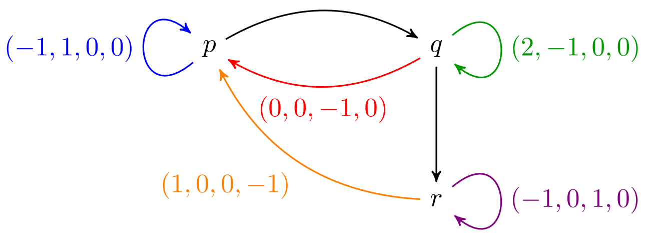
$$p_1(1,0,n) \longrightarrow \ldots \longrightarrow q_1(2^n,0,0) \longrightarrow r(2^n,0,0) \longrightarrow \ldots \longrightarrow r(0,0,2^n)$$
$$\longrightarrow p_2(1,0,2^n) \longrightarrow \ldots q_2(2^{2^n},0,0).$$

In other words in the first copy of the HP-gadget from $p_1(1,0,n)$ we reach $p_2(2^n,0,0)$. Then in state r we transfer value from the third counter to the first one. The transition from r to p_2 adds one to the first counter such that we start from $p_2(1,0,2^n)$ in the second copy of the HP-gadget.

It is easy to show that the reachability set of $p_1(1,0,n)$ is finite, the proof goes as in Example 1.

In a similar way one can constructs a 3-VASS which has k-fold exponential reachability set, we just take k copies of the HP-gadget and connect them by states r_i as above. However this requires a growing number of states in a VASS. Here comes another idea: by adding just one additional counter we can simulate any number of copies on this counter.

▶ **Example 3.** The following 4-VASS has finite, but tower size reachability set. It is just a slight modification of Example 2.



In this 4-VASS we have added the fourth counter and the only transition which modifies this counter is the orange transition. The rest is exactly like in the HP gadget with additional state r. Thus for any k we have to following run:

$$p(1,0,k,n) \longrightarrow \ldots \longrightarrow q(2^k,0,0,n) \longrightarrow r(2^k,0,0,n) \longrightarrow \ldots \longrightarrow r(0,0,2^k,n) \longrightarrow p(1,0,2^k,n-1).$$

In other words we can exponentiate the first counter for the cost of decreasing the forth counter by one. Thus for any $n \in \mathbb{N}$ there is also the following run:

$$p(1,0,1,n) \longrightarrow p(2,0,1,n-1) \longrightarrow p(4,0,1,n-2) \longrightarrow \ldots \longrightarrow p(\text{Tower}(n),0,1,0).$$

This easily implies that the reachability set from $p(1,0,1,n)$ is of at least Tower(n) size. It remains to show that this reachability set is finite. To see this notice first that the orange transition can be fired at most n times. Now it is easy to see that in between of any two firings of the orange transition other transitions can be fired at most exponentially many times wrt. the current counter values, which finishes the argument.

The Example 3 already shows that a very simple VASS can have a pretty complicated behaviour. It is not hard to see that in a similar vein one can construct in any dimension d a unary d-VASS with finite reachability set of size around $F_{d-1}(n)$, where n is the size of the source configuration.

5 Finite reachability sets are enough

It is a good moment to emphasise that authors of [16] not only have shown that the reachability problem in d-VASSes can be solved in $\mathcal{F}_{d+4}$, but they proved that if there is a run from the source to the target then there is also one of length bounded by roughly speaking $F_{d+4}(n)$. Using this result and the generalised Example 3 one can show that VASSes with finite reachability sets are actually not much simpler than VASSes without that restriction. More concretely speaking one can reduce the reachability problem for d-VASSes to the reachability problem for $(d+6)$-VASSes with finite reachability sets. Assume we need to check whether $s \longrightarrow t$ in a d-VASS V. We construct a $(d+6)$-VASS U as follows. First part of U behaves like generalised Example 3 in dimension $d+5$, thus on one of the counters (say counter number $d+5$) can have values up to $F_{d+4}(n)$. We use the last $(d+6)$-th counter to keep the sum of all the dimensions numbered from 1 to $d+4$. In the second part U simulates V on dimensions from 1 to d. In the target configuration of U we demand that dimension $d+6$ is equal to zero, so after the first part all the dimensions from 1 to $d+4$ need also to be zero. The only change of the second part of U wrt. to V is that to simulate any transition of V in U we decrease the $(d+5)$-th counter by one. Notice now that if there is a run from s to t in U by [16] there is also one of length at most $F_{d+4}(n)$ thus there is also one in V. Of course no run in U implies no run in V as the simulation is faithful. On the other hand the reachability set of any configuration in V is finite as in each step we decrease the $(d+5)$-th counter. This finishes the argument.

The above reasoning does not show that considering VASSes with finite reachability sets is enough, because we have added six additional dimensions. However it suggests that in order to understand well low dimensions it might be sufficient to look sometimes at this special case of finite reachability set. Notice that this is a strong statement, as the reachability problem can be easily solved for VASSes with finite reachability set: we just compute the whole set of configurations reachable from the source and after this computation stops (it has to, as the reachability set is finite) we check whether the target belongs to the set. Moreover we have a pretty good complexity upper bounds for this very naive algorithm. By [7] the longest sequence of configurations in a d-VASS without a *domination* (situation that a configuration further in the sequence is strictly bigger than a configuration earlier in the sequence) is bounded roughly speaking by $F_{d+1}(n)$, where n upper bounds the size of VASS and the source configuration. Notice that in VASSes with finite reachability set no run has a domination, as domination allows for pumping counters up and would imply an infinite reachability set. Thus [7] shows that exploring the whole space of reachable configurations in a d-VASS can be achieved in the complexity class $\mathcal{F}_{d+1}$.

Notice however that for 3-VASSes even assuming finite reachability set we still get complexity $\mathcal{F}_4$, which is much higher than the known lower bound of PSpace-hardness. Thus there might be a possibility of constructing a 3-VASS or other lower dimensional VASS with shortest run being exponential, doubly exponential or even Tower length. Below we show a few current, still very weak, techniques which can lead in the future to some involved examples.

6 Telescope equations

Example 3 and its generalisations exhibit a complicated behaviour of low dimensional VASSes. Notice however that it does not eliminate a possibility that in low dimensional VASSes there are always some short paths. Imagine the following slight modification of Example 3.

$$\begin{tikzcd}
 & p \arrow[loop left, "{(-1,1,0,0)}"] \arrow[r, bend left] \arrow[dl, bend left] & q \arrow[loop right, "{(2,-1,0,0)}"] \arrow[l, bend left, "{(0,0,-1,0)}"] \arrow[d] \\
t \arrow[loop left, "{(-1,0,0,0)}"] & & r \arrow[loop right, "{(-1,0,1,0)}"] \arrow[ul, bend left, "{(1,0,0,-1)}"]
\end{tikzcd}$$

From Example 3 we know that in the above VASS there are Tower(n)-long paths from $p(1,0,1,n)$ to $t(0,0,1,0)$: such a path first reaches $p(\text{Tower}(n),0,1,0)$, then goes to $t(\text{Tower}(n),0,1,0)$ and then in a loop decreases the first counter. However there are also some very short runs: we n times apply the sequence

$$p(\ell,0,1,k) \longrightarrow q(\ell,0,1,k) \longrightarrow r(\ell,0,1,k) \longrightarrow p(\ell+1,0,1,k-1)$$

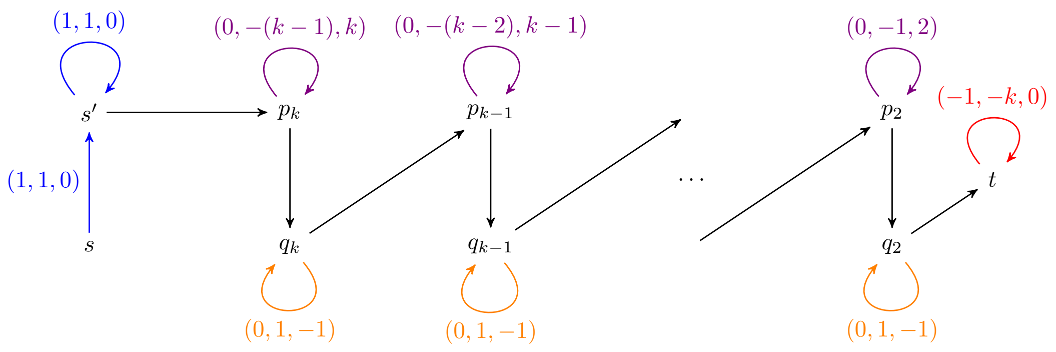
then go to state t and quickly decrease the first counter.

This illustrates the main challenge with proving lower bounds for the reachability problem in VASSes: it is very hard to force a VASS to take some long run from the source to the target. Here we present one approach how to force a VASS to have only long runs, the example is taken from [3]. It is based on the following simple telescope equation:

$$k = \frac{k}{k-1} \cdot \frac{k-1}{k-2} \cdot \ldots \cdot \frac{3}{2} \cdot \frac{2}{1}. \qquad (1)$$

Based on (1) we build a 3-VASS V_k with size of all the transitions bounded by k and a property that the shortest path from the source to the target is of length exponential in k.

▶ **Example 4.** In this example the source configuration is $s(0,0,0)$ and the target configuration is $t(0,0,0)$.



Let us analyze how a run from the source to the target in VASS V_k can look like. In the state s we fire the blue transition to $s'(1,1,0)$ and then some number $N-1$ times the blue loop in state s' reaching the configuration $s'(N,N,0)$. So the prefix of our run is the following

$$s(0,0,0) \longrightarrow s'(1,1,0) \longrightarrow \ldots \longrightarrow s(N,N,0) \longrightarrow p_k(N,N,0).$$

States s and s' are distinguished to assure that $N \geq 1$. Notice now that the only other transition in V_k which modifies the first counter is the red transition in state t. Thus the considered run need to finish in the following way:

$$q_2(N,Nk,0) \longrightarrow t(N,Nk,0) \longrightarrow \ldots \longrightarrow t(0,0,0).$$

Observe now that for each $i \in \{2,\ldots,k\}$ the orange transitions in q_i do not change the sum of the second and the third counter while the violet transitions in p_i can multiply this sum by at most $\frac{i}{i-1}$. Moreover this is the case if and only if the run enters p_i in the configuration of the form $p_i(0,K,0)$ where K is divisible by $i-1$ and leaves it in the configuration of the form $p_i(0,0,K \cdot \frac{i}{i-1})$. In other words in the state p_i the whole value of second counter needs to be transferred to the third counter while multiplying it by $\frac{i}{i-1}$. Notice now that from $p_k(N,N,0)$ till $q_2(N,Nk,0)$ the second counter needs to be multiplied by exactly k. Using Equation (1) we derive that in any run from $p_k(N,N,0)$ to $q_2(N,Nk,0)$ in all the states the whole value of the second counter have to be transferred to the third counter or vice versa. In particular each loop have to be fired the maximal number of times. So the run needs to look as follows:

$$\begin{aligned}
p_k(N,N,0) &\longrightarrow p_k(N,0,\frac{Nk}{k-1}) \longrightarrow q_k(N,0,\frac{Nk}{k-1}) \longrightarrow q_k(N,\frac{Nk}{k-1},0) \longrightarrow p_{k-1}(N,\frac{Nk}{k-1},0)\\
&\longrightarrow p_{k-1}(N,0,\frac{Nk}{k-2}) \longrightarrow q_{k-1}(N,0,\frac{Nk}{k-2}) \longrightarrow q_{k-1}(N,\frac{Nk}{k-2},0) \longrightarrow p_{k-2}(N,\frac{Nk}{k-2},0)\\
&\dots\\
&\longrightarrow p_3(N,0,\frac{Nk}{2}) \longrightarrow q_3(N,0,\frac{Nk}{2}) \longrightarrow q_3(N,\frac{Nk}{2},0) \longrightarrow p_2(N,\frac{Nk}{2},0)\\
&\longrightarrow p_2(N,0,Nk) \longrightarrow q_2(N,0,Nk) \longrightarrow q_2(N,Nk,0).
\end{aligned}$$

Now notice that in the run for each $i \in \{2,\dots,k-1\}$ we have a configuration $q_i(N,\frac{Nk}{i},0)$, which means that Nk is divisible by each $i \in \{2,\dots,k-1\}$. Thus Nk is a multiplicity of the $\mathrm{lcm}(2,\dots,k-1)$, which is known to be exponential wrt. k (see [3], Claim 6). This finishes the proof that any run from the source to the target needs to be of length exponential wrt. k.

The above example can also be expressed by another formalism, which is often much more convenient to present VASSes then drawing them as automata. This formalism is called the counter programs. We do not introduce counter programs formally, instead we present VASS from Example 4 as a counter program hoping that this clarifies the issue. We assume that the three counters are named x, y and z. For more details look into [3].

```
$x$ += 1    $y$ += 1
loop
    $x$ += 1    $y$ += 1
for $i$ := $k$ down to 2 do
    loop
        $y$ −= $i-1$    $z$ += $i$
    loop
        $y$ += 1    $z$ −= 1
loop
    $x$ −= 1    $y$ −= $k$
```

Using similar trick with the telescope equation (but a bit more involved) we have shown in [3] an example a 4-VASS in which the shortest run from the source to the target is of doubly-exponential length.

7 Controlling-counter

VASSes, in contrast to counter machines lack zero-tests, thus it is pretty hard to force their runs to be exact. Notice that with zero-tests we can easily force the modified Example 3 (mentioned in paragraph Telescopic equations) to have only runs of Tower length. We just enforce that all the loops are fired maximally by zero-testing appropriate counters after the loops. Of course we cannot hope to simulate zero-tests by VASSes as VASSes with zero-tests (called counter machines) have undecidable reachability problem.

However, we are able to simulate some restricted number of zero-tests in VASSes. First of all notice that in the reachability problem we ask whether we can reach the target configuration, so we already have some very weak for of zero-tests: if we set the target configuration to be zero at some counter then we can test this counter to be zero at the end of the run. Now the idea is to boost this single zero-test to simulate more zero-tests during the run.

Let us assume that we have a d-VASS V with some the counter x and we want to zero-test counter x in some three moments during the run. First very naive idea is to add three additional counters x_1, x_2, x_3 to V, which are copies of x and modify them exactly as x. The first one is stopped being modified after the first moment, the second one is not modified after the second moment and the third one is not modified after the third moment. In this way if in the modified $(d+3)$-VASS we set the target configuration to be zero on counters x_1, x_2, x_3 then we enforce that any run reaching the target indeed have value zero in the three considered moments. The main drawback of this idea is that it introduces additional counters, so is too costly. However, already this technique illustrates that zero-test in the target configuration can be used to simulate zero-tests in other moments in the run.

Here we introduce the technique of the controlling-counter, which was proposed in [4]. Assume we have a run ρ in our d-VASS V of the following form:

$$s \xrightarrow{\rho_1} c_1 \xrightarrow{\rho_2} c_2 \xrightarrow{\rho_3} c_3 \xrightarrow{\rho_4} t$$

and we want to zero-test the counter x in the configurations c_1, c_2, c_3. Let us assume that the value of the counter x in the source configuration s is zero. Let the value of the counter x in configurations c_i be x_i, for $i \in \{1, 2, 3\}$. We need to check whether $x_1 = x_2 = x_3 = 0$. Notice that it is enough to check if $x_1 + x_2 + x_3 = 0$ as all the counter values x_i are nonnegative. Let Δ_i be the effect of the run ρ_i on the counter x. Thus we have $x_1 = \Delta_1$, $x_2 = \Delta_1 + \Delta_2$ and $x_3 = \Delta_1 + \Delta_2 + \Delta_3$. Therefore $x_1 + x_2 + x_3 = 3\Delta_1 + 2\Delta_2 + \Delta_3$ and it is enough to check whether this expression has value zero. In order to do that we introduce one additional *controlling-counter* y which is tested for zero in the target configuration t. We set the value of the counter y in the configuration s to be zero. Each change of x by C in ρ_1 is matched by change of y by $3C$. Similarly, each change of x by C in ρ_2 is matched by change of y by $2C$. Finally, each change of x by C in ρ_3 is matched by change of y by the same value C. Thus indeed final value of y is exactly $3\Delta_1 + 2\Delta_2 + \Delta_3$ and it is enough to check y for zero in the target configuration in order to assure that $x_1 = x_2 = x_3 = 0$.

It is easy to observe that this reasoning can be extended to any number of zero-tests. In general if we are in the part of the run ρ such that after this part still k zero-tests are performed on x then each change of x by C needs to be matched by the change of y by $k \cdot C$. We only need that configurations $c_1, c_2, c_3, \ldots$ are distinguishable in the sense that we can change behaviour of counter y after any c_i. This can be often easily implemented by use of states.

It is also not hard to see that one controlling-counter can control many original counters, not just one.

Below we present the simplest possible application of the controlling-counter to 3-VASSes. Consider the following 2-VASS with two counters x and y starting in the counter valuation $(x, y) = (1, 0)$.

```
1: for $i$ := 1 to $k$ do
2:     loop
3:         $x$ −= 1   $y$ += 2
4:     loop
5:         $x$ += 1   $y$ −= 1
6: loop
7:     $x$ −= 1
```

It is easy to see that if all the loops are fired maximally then before entering line 6 counter values are $(x, y) = (2^k, 0)$ and loop in lines 6-7 can be fired 2^k times. Thus if we want to reach values $(0, 0)$ at the end of the counter program there exists an exponential run. However, there is also a very short run, the one totally ignoring the loops in lines 2-3 and in lines 4-5 and immediately jumping to the loop in lines 6-7 which is fired just once. However, introducing a controlling-counter z we may enforce the loops to be fired maximal number of times and thus obtain another example of a VASS with shortest one run being exponential.

▶ **Example 5.** In the 2-VASS above both counters x and y are tested exactly k times. Thus as the starting valuation is $(x, y) = (1, 0)$ we should start from value $z = k$. Therefore in the i-th iteration of the for-loop in the line 3 the counter x is still waiting for $k - (i - 1)$ zero-tests as well as the counter y. Similarly as in the line 3 the counter x in the line 5 is still waiting for $k - i$ zero-tests while the counter y is waiting for $k - (i - 1)$ zero-tests. Therefore in the line 3 we should increase z by $(-1) \cdot (k - i + 1) + 2 \cdot (k - i + 1) = k - i + 1$ while in the line 5 we should increase z by $1 \cdot (k - i) + (-1) \cdot (k - i + 1) = -1$. Therefore the resulting 3-VASS have the property that the shortest (and the only) run from $(1, 0, k)$ to $(0, 0, 0)$ is exponential in k.

```
1: for $i$ := 1 to $k$ do
2:     loop
3:         $x$ −= 1   $y$ += 2   $z$ += $k - i + 1$
4:     loop
5:         $x$ += 1   $y$ −= 1   $z$ −= 1
6: loop
7:     $x$ −= 1
```

The use of the controlling-counter technique may be much more intricate, however the above Example 5 presents its main idea. In [4] the whole Ackermann-hardness idea was based on controlling-counters. Lasota in [12] simplified our approach and presented it without the use of controlling-counters. It turns however that in low dimensions controlling-counter technique can be very convenient as it uses only one additional dimension to control others in contrast to the multiplication triple technique (explained below), which requires three dimensions (at least in its classical version). Below we briefly describe the multiplication triple technique. We also show how to use it together with the controlling-counter technique to obtain Tower-hardness for the reachability problem in VASSes already in dimension eight.

8 Multiplication triples

Both the above presented techniques of telescope equations and controlling-counter are useful for designing VASSes with long runs, but it is not clear how they solely can be used to get some complexity lower bounds.

Here we briefly introduce the technique of multiplication triples and show how to use it to get pretty easily PSpace-hardness lower bound for the reachability problem in unary 7-VASSes. Notice that it is not hard to improve this result, in [4] we have show PSpace-hardness for unary 5-VASSes. Here we present this simple result to illustrate briefly an application of the multiplication triple technique.

Recall that a d-counter machine is just a d-VASS with possibility of zero-tests. We say that a run of a counter machine is B-bounded if at each configuration on this run the sum of all the counter values does not exceed B. We first recall the following theorem, which is a folklore.

▶ **Theorem 6.** *The problem whether a given three-counter machine for a given number $n \in \mathbb{N}$ has a 2^n-bounded run from a given source configuration to a given target configuration is* PSpace-*hard.*

The main idea behind the multiplication triple technique is that a d-VASS equipped with three additional counters (x, y, z) with initial values (B, C, BC) can simulate $C/2$ zero-tests on B-bounded counters. Here we do not explain how this simulation exactly works and why this is the case, explanations can be found in [12, 4]. It is important for us here that in order to obtain PSpace-hardness for 7-VASSes it is enough to construct a family V_n of 7-VASSes with the following properties:

- transition values of V_n are bounded by $2n$ (any polynomial function of n is fine),
- all the reachable configurations of the form $t(x_1, \ldots, x_7)$, where t is the target state, have the property that if $x_7 = 0$ then $x_1 = x_2 = x_3 = 0$, $x_4 = 2^n$ and $x_6 = 2^n \cdot x_5$.

Intuitively speaking by testing x_7 for zero in the target configuration we get a triple of the form $(2^n, C, 2^n \cdot C)$ on counters (x_4, x_5, x_6). In the latter part of the VASS run we can simulate a three-counter machine on counters (x_1, x_2, x_3) and use the counters (x_4, x_5, x_6) to check whether x_1, x_2, x_3 are indeed 2^n-bounded and for simulating zero-tests on them. Thus in the rest of this paragraph we focus on showing how to construct the above family V_n. Recall that counter programs are just ways of presenting VASSes, so we interchangeably speak about VASSes and counter programs.

▶ **Example 7.** The idea is simple. We only use counters x_1, x_4, x_5, x_6, x_7. We first set $x_4 = 1$ and $x_5 = x_6 = C$ for some guessed value C. Then using x_1 as an auxiliary counter we multiply n times counters x_4 and x_6 by 2. Counter x_7 is used as the controlling-counter to assure that the multiplications are exact. During this process the counters x_4 and x_6 are zero-tested n times while the counter x_1 is zero-tested $2n$ times. Therefore in the line 1 the increase of x_4 by 1 results in the increase of x_7 by n. Similarly in line 3 the increase of x_6 by 1 results in the increase of x_7 by $2n$. In the i-th iteration of the for-loop we have that:

- in the line 6 counter x_4 is waiting for $n - i + 1$ zero-tests and counter x_1 is waiting for $2(n - i + 1)$ zero-tests, so x_7 should be increased by $3n - 3i + 3$,
- in the line 8 counter x_1 is waiting for $2(n - i + 1)$ zero-tests and counter x_4 is waiting for $n - i$ zero-tests, so x_7 should be decreased by $n - i + 2$,
- in the line 6 counter x_6 is waiting for $n - i + 1$ zero-tests and counter x_1 is waiting for $2(n - i + 1) - 1$ zero-tests, so x_7 should be increased by $3n - 3i + 1$,
- in the line 8 counter x_1 is waiting for $2(n - i + 1) - 1$ zero-tests and counter x_6 is waiting for $n - i$ zero-tests, so x_7 should be decreased by $n - i + 1$,

```
1:  $x_4$ += 1    $x_7$ += $n$
2:  loop
3:      $x_5$ += 1    $x_6$ += 1    $x_7$ += $2n$
4:  for $i$ := 1 to $n$ do
5:      loop
6:          $x_4$ −= 1    $x_1$ += 2    $x_7$ += $3n - 3i + 3$
7:      loop
8:          $x_1$ −= 1    $x_4$ += 1    $x_7$ −= $n - i + 2$
9:      loop
10:         $x_6$ −= 1    $x_1$ += 2    $x_7$ += $3n - 3i + 1$
11:     loop
12:         $x_1$ −= 1    $x_6$ += 1    $x_7$ −= $n - i + 1$
```

If after this counter program the controlling-counter x_7 has value zero then it means that indeed $x_1 = 0$, $x_4 = 2^n$, $x_6 = 2^n \cdot x_5$ and clearly $x_2 = x_3 = 0$, so all the necessary conditions for PSpace-hardness are fulfilled.

The above example shows how to join forces of controlling-counter and multiplication triples technique to rather easily show some not entirely trivial PSpace-hardness lower bound for 7-VASSes. By more clever constructions we can get a bit stronger lower bounds, but we are still very far away from matching the upper and the lower bounds for the reachability problem in low dimensional VASSes.

9 Afterthought

In this short tutorial we tried to present in the simplest possible way almost the whole spectrum of current techniques of designing involved VASSes. Many of the applications are more elaborate than the presented once, however it is still surprising that most of them are not extremely complicated and some problems open for decades are solvable by techniques which are at the end of the day rather simple. In my opinion we still need at least a few more techniques in order to understand what phenomena are hiding in the low dimensional VASSes.

References

1 Michael Blondin, Alain Finkel, Stefan Göller, Christoph Haase, and Pierre McKenzie. Reachability in two-dimensional vector addition systems with states is PSpace-complete. In *Proceedings of LICS 2015*, pages 32–43, 2015.

2 Wojciech Czerwinski, Slawomir Lasota, Ranko Lazic, Jérôme Leroux, and Filip Mazowiecki. The reachability problem for Petri nets is not elementary. In *Proceedings of STOC 2019*, pages 24–33. ACM, 2019.

3 Wojciech Czerwinski, Slawomir Lasota, Ranko Lazic, Jérôme Leroux, and Filip Mazowiecki. Reachability in fixed dimension vector addition systems with states. In *Proceedings of CONCUR 2020*, pages 48:1–48:21, 2020.

4 Wojciech Czerwinski and Lukasz Orlikowski. Reachability in vector addition systems is Ackermann-complete. In *Proceedings of FOCS 2021*, pages 1229–1240, 2021.

5 Wojciech Czerwinski and Lukasz Orlikowski. Lower bounds for the reachability problem in fixed dimensional vasses. *CoRR*, abs/2203.04243, 2022.

6 Matthias Englert, Ranko Lazic, and Patrick Totzke. Reachability in two-dimensional unary vector addition systems with states is NL-complete. In *Proceedings of LICS 2016*, pages 477–484, 2016.

7 Diego Figueira, Santiago Figueira, Sylvain Schmitz, and Philippe Schnoebelen. Ackermannian and Primitive-Recursive Bounds with Dickson's Lemma. In *Proceedings of LICS 2011*, pages 269–278, 2011.

8 Christoph Haase, Stephan Kreutzer, Joël Ouaknine, and James Worrell. Reachability in succinct and parametric one-counter automata. In *Proceedings of CONCUR 2009*, pages 369–383, 2009.

9 John E. Hopcroft and Jean-Jacques Pansiot. On the reachability problem for 5-dimensional vector addition systems. *Theor. Comput. Sci.*, 8:135–159, 1979.

10 S. Rao Kosaraju. Decidability of reachability in vector addition systems (preliminary version). In *Proceedings of STOC 1982*, pages 267–281, 1982.

11 Jean-Luc Lambert. A structure to decide reachability in Petri nets. *Theor. Comput. Sci.*, 99(1):79–104, 1992.

12 Slawomir Lasota. Improved Ackermannian Lower Bound for the Petri Nets Reachability Problem. In *Proceedings of STACS 2022*, volume 219 of *LIPIcs*, pages 46:1–46:15, 2022.

13 Jérôme Leroux. The reachability problem for petri nets is not primitive recursive. In *Proceedings of FOCS 2021*, pages 1241–1252, 2021.

14 Jérôme Leroux. The reachability problem for petri nets is not primitive recursive. *CoRR*, abs/2104.12695, 2021.

15 Jérôme Leroux and Sylvain Schmitz. Demystifying reachability in vector addition systems. In *Proceedings of LICS 2015*, pages 56–67, 2015.

16 Jérôme Leroux and Sylvain Schmitz. Reachability in vector addition systems is primitive-recursive in fixed dimension. In *Proceedings of LICS 2019*, pages 1–13. IEEE, 2019.

17 Richard J. Lipton. The reachability problem requires exponential space. Technical report, Yale University, 1976.

18 Ernst W. Mayr. An algorithm for the general Petri net reachability problem. In *Proceedings of STOC 1981*, pages 238–246, 1981.

19 Charles Rackoff. The covering and boundedness problems for vector addition systems. *Theor. Comput. Sci.*, 6:223–231, 1978.

20 Sylvain Schmitz. Complexity hierarchies beyond elementary. *ACM Trans. Comput. Theory*, 8(1):3:1–3:36, 2016.

21 Richard Zurawski and MengChu Zhou. Petri nets and industrial applications: A tutorial. *IEEE Trans. Ind. Electron.*, 41(6):567–583, 1994.