Z-Reachability Problem for Games on 2-Dimensional Vector Addition Systems with States Is in P[⋆]

Jakub Chaloupka

Faculty of Informatics, Masaryk University,
Botanická 68a, 60200 Brno, Czech Republic
`xchalou1@fi.muni.cz`

Abstract. We consider a two-player infinite game with zero-reachability objectives played on a 2-dimensional vector addition system with states (VASS), the states of which are divided between the two players. Brázdil, Jančar, and Kučera (2010) have shown that for $k > 0$, deciding the winner in a game on k-dimensional VASS is in $(k-1)$-$\mathsf{EXPTIME}$. In this paper, we show that, for $k = 2$, the problem is in P, and thus improve the $\mathsf{EXPTIME}$ upper bound.

1 Introduction

Vector addition systems with states (VASS) are an abstract computational model equivalent to Petri nets [6] which is well suited for modelling and analysis of distributed concurrent systems. Roughly speaking, a *k-dimensional VASS*, where $k > 0$ is an automaton with a finite control and k unbounded counters which can store non-negative integers. It can be represented as a finite k-weighted directed graph $G = (V, E, w)$. For simplicity, the weights of the edges are restricted to vectors from the set $\{-1, 0, 1\}^k$. At the beginning of the computation, a token is placed on one of the vertices. In each step of the computation, a VASS can move the token to one of the destination vertices of the edges emanating from the current vertex with the token. This also updates the vector of current counter values by adding the weight of the traversed edge. Since the counters cannot become negative, transitions which attempt to decrease a zero counter are disabled. Configurations of a given VASS are written as pairs $(v, \vec{n})$, where v is the current vertex and $\vec{n} \in \mathbb{N}_0^k$ is a vector of the current counter values.

Brázdil, Jančar, and Kučera [1] extended VASS in two respects. First, the set of vertices is divided between two players, named $\Box$ and $\Diamond$, and so we get a turn-based two-player game where the choice of an outgoing edge is upon the player who owns the current vertex with the token. Second, the weights of edges may contain symbolic components (denoted by ω) whose intuitive meaning is "add an arbitrarily large non-negative integer to the appropriate counter". Edges

[⋆] This work has been partially supported by the Grant Agency of the Czech Republic grants No. 201/09/1389, 102/09/H042.

with symbolic components represent infinite number of transitions. This two-fold extension of a VASS is called a *game on k-dim VASS* and it has been shown in [1] to be capable of modelling interesting systems.

Various problems on games on k-dim VASS have been considered in [1]. In particular, the Z-reachability problem is the problem of deciding whether for a given starting configuration $(v, \overrightarrow{n})$, the player $\Box$ has a strategy that ensures that not one of the k counters is ever equal zero, which is the complement of the problem of deciding whether the player $\Diamond$ has a strategy that ensures that eventually at least one of the counters is zero, i.e., a configuration $(v', (n'_1, \ldots, n'_k))$ such that $(\exists i \in \{1, \ldots, k\})(n'_i = 0)$ is reached. This problem was shown in [1] to belong to the complexity class $(k-1)$-EXPTIME. In particular, for $k = 1$ and $k = 2$, the problem is in P and EXPTIME, respectively.

Our Contribution. In this paper, we show that 2-dimensional VASS games with Z-reachability objectives are solvable in polynomial time, and thus improve the EXPTIME upper bound given in [1]. More precisely, we show that the winner in 2-dim VASS games can be decided in polynomial time, and a finite description of winning starting configurations of both players is also computable in polynomial time. This contrasts sharply with the previous results about VASS (or, equivalently, Petri nets) where the undecidability/intractability border usually lies between one and two counters. For example, k-dim VASS are equivalent to Petri nets with k unbounded places, and it has been shown that the bisimilarity problem is decidable for Petri nets with one unbounded place and undecidable for Petri nets with two or more unbounded places [4,5]. The Z-reachability problem for games on 2-dim VASS also seems to be harder than the 1-dim case, because unlike for the games on 1-dim VASS, in games on 2-dim VASS, if we add an arbitrarily small rational number to some element of some edge-weight, then the set of vertices of G which are part of some winning configuration for $\Box$ may change.

An interesting open question is whether the techniques presented in this paper can be extended to three- (or even more-) dimensional VASS games. Since the presented results about 2-dimensional VASS are relatively complicated (despite investing some effort, we did not manage to find any substantial simplifications), we suspect this problem as difficult.

The Z-reachability problem for games on k-dim VASS can be also thought of as a problem of deciding the winner in an ordinary two-player reachability game with infinite arena. The arena consists of all possible configurations $(v, \overrightarrow{n}) \in V \times \mathbb{N}_0^k$ and it is divided between $\Box$ and $\Diamond$ according to the first component of the configurations. The set of target configurations is the set $\mathsf{Z} = \{(v, (n_1, \ldots, n_k)) \mid (\exists i \in \{1, \ldots, k\})(n_i = 0)\}$. $\Box$ wants to avoid the set Z while $\Diamond$ wants to reach it. We note that the game is upward-closed in the sense that if $\Box$ has a strategy to win from a configuration $(v, \overrightarrow{n}) \in V \times \mathbb{N}_0^k$, then the same strategy also wins each play starting from $(v, \overrightarrow{n}') \in V \times \mathbb{N}_0^k$ such that $\overrightarrow{n}' \geq \overrightarrow{n}$. Therefore, there is a finite set of minimal winning starting configurations.

2 Preliminaries

For technical convenience, we will define the game in a slightly different way than in Section 1, and then we will show how the properties of the modified game imply existence of a polynomial algorithm for solving the original game. The properties of the modified game are proved in Section 3, the main part of this paper.

A *game on 2-dim vector addition system with states (VASS)* is a tuple $\Gamma = (G, V_\Box, V_\Diamond)$, where $G = (V, E, w)$ is a finite two-weighted directed graph such that V is a disjoint union of the sets $V_\Box$ and $V_\Diamond$, $E \subseteq V^2$, $w : E \to \{-1, 0, 1\}^2$, and each vertex has at least one outgoing edge. The graph G can also be thought of as a 2-dim VASS [3]. The game is played by two opposing players, named $\Box$ and $\Diamond$. A play starts by placing a token on some given vertex and the players move the token along the edges of G ad infinitum. If the token is on vertex $v \in V_\Box$, $\Box$ moves it. If the token is on vertex $v \in V_\Diamond$, $\Diamond$ moves it. This way an infinite path $p_\infty = (v_0, v_1, v_2, \ldots)$ is formed. The path p_∞ is also called a *play*. The play is winning for $\Box$, if both components of the sum of the weights of the traversed edges are above some constant $K \in \mathbb{Z}$ during the whole play, i.e., $(\exists K \in \mathbb{Z})(\forall k \in \mathbb{N}_0)(\sum_{i=0}^{k-1} w(v_i, v_{i+1}) \geq (K, K))$ where the sum and the inequality are element-wise. The play is winning for $\Diamond$, if for any constant $K \in \mathbb{Z}$, there is a point in the play where at least one of the components of the sum of the traversed edges is below K, i.e., $(\forall K \in \mathbb{Z})(\exists k \in \mathbb{N}_0)(\sum_{i=0}^{k-1} w_1(v_i, v_{i+1}) < K \vee \sum_{i=0}^{k-1} w_2(v_i, v_{i+1}) < K)$. Please note that the initial vector of counter values is $(0, 0)$ and the counters are allowed to go negative.

A *strategy* of $\Box$ is a function $\sigma : V^* \cdot V_\Box \to V$ such that for each finite path $p = (v_0, \ldots, v_k)$ with $v_k \in V_\Box$, it holds that $(v_k, \sigma(p)) \in E$. Recall that each vertex has out-degree at least one, and so the definition of a strategy is correct. The set of all strategies of $\Box$ in Γ is denoted by Σ^Γ. We say that an infinite path $p_\infty = (v_0, v_1, v_2, \ldots)$ agrees with the strategy $\sigma \in \Sigma^\Gamma$ if for each $v_i \in V_\Box$, $\sigma(v_0, \ldots, v_i) = v_{i+1}$. A strategy π of Min is defined analogously. The set of all strategies of Min in Γ is denoted by Π^Γ. Given an initial vertex $v \in V$, the *outcome* of two strategies $\sigma \in \Sigma^\Gamma$ and $\pi \in \Pi^\Gamma$ is the (unique) infinite path $\mathsf{outcome}^\Gamma(v, \sigma, \pi) = (v = v_0, v_1, v_2, \ldots)$ that agrees with both σ and π.

The set V can be partitioned into two sets, $W_\Box$ and $W_\Diamond$, so that if the play starts at some vertex $v \in W_\Box$, then $\Box$ has a strategy that ensures that he will win, and if the play starts at some vertex $v \in W_\Diamond$, then $\Diamond$ has a strategy that ensures that she will win [1]. Formally:

$$\begin{aligned} v \in W_\Box \Leftrightarrow\ & (\exists \sigma \in \Sigma^\Gamma)(\forall \pi \in \Pi^\Gamma) \\ & (\ \mathsf{outcome}^\Gamma(v, \sigma, \pi) = (v = v_0, v_1, v_2, \ldots) \wedge \\ & \quad (\exists K \in \mathbb{Z})(\forall k \in \mathbb{N}_0)(\textstyle\sum_{i=0}^{k-1} w(v_i, v_{i+1}) \geq (K, K))\ ) \end{aligned} \tag{1}$$

To solve the game is to determine the sets $W_\Box$ and $W_\Diamond$. In this paper, we will show that there is a constant $K_{\min} \in \mathbb{Z}$ of *polynomial size* with respect to $|V|$ such that for each $v \in W_\Box$, the constant K in (1) can always be chosen so that $K \geq K_{\min}$. By the statement that $K_{\min} \in \mathbb{Z}$ is of polynomial size with respect to $|V|$, we mean that $|K_{\min}| \leq l \cdot |V|^k$ for some fixed constants $k, l \in \mathbb{N}$.

The polynomial size of $K_{\min}$ implies that the values of both counters in all minimal winning configurations of $\Box$ in the original reachability game with infinite arena is of polynomial size with respect to $|V|$ (cf. the full version of this paper [2]). It follows that we can obtain the solution of the original game by solving only a restricted game, where the values of both counters are bounded by a number of polynomial size with respect to $|V|$. Since a reachability game can be solved in polynomial time with respect to the number of its configurations, we have a polynomial-time algorithm for solving the original reachability game with infinite arena. Our definition of the game on 2-dim VASS does not consider edge-weights with the symbolic component ω. We outline how to extend the proofs to games with symbolic components in edge-weights in the full version of this paper [2].

If $e \in E$, then $w_1(e)$ is the first component of $w(e)$ and $w_2(e)$ is the second component of $w(e)$, i.e., $w(e) = (w_1(e), w_2(e))$. Simple cycle in G is a cycle with no repeated vertex. In this paper, we will work only with simple cycles, and so we will often omit the adjective "simple". If $c = (v_0, \ldots, v_{k-1}, v_k = v_0)$ is a cycle, then $w(c)$ is the sum of the weights of its edges, element-wise, i.e., $w(c) = (\sum_{i=0}^{k-1} w_1(v_i, v_{i+1}), \sum_{i=0}^{k-1} w_2(v_i, v_{i+1}))$. The terms $w_1(c)$ and $w_2(c)$ have the intuitive meaning. Because of the limitations on the weights of the edges, it always holds that $|w_1(c)|, |w_2(c)| \leq |V|$, for each cycle c in G. The weight of a path $(v_0, \ldots, v_k)$ is defined analogically.

The cycles of G can be partitioned into four sets. The first set, P, is the set of cycles c such that $w_1(c) \geq 0 \wedge w_2(c) \geq 0$. The second set, N, is the set of cycles c such that $(w_1(c) \leq 0 \wedge w_2(c) < 0) \vee (w_1(c) < 0 \wedge w_2(c) \leq 0)$. The third set, A, is the set of cycles c such that $w_1(c) > 0 \wedge w_2(c) < 0$. Finally, the fourth set, B, is the set of cycles such that $w_1(c) < 0 \wedge w_2(c) > 0$.

The ratio of the weights of a cycle c is the fraction $\frac{w_1(c)}{w_2(c)}$. We will use $\mathcal{R}$ to denote the set of all possible ratios of weights of the cycles from $A \cup B$, i.e., $\mathcal{R} = \{\frac{a}{b} \mid a \in \{-|V|, \ldots, -1\} \wedge b \in \{1, \ldots, |V|\}\}$. For each $X \in \{A, B\}$, $\sim \in \{<, \leq, =, \geq, >\}$, and $R \in \mathcal{R}$, we will use $X_{\sim R}$ to denote the set of cycles $\{c \in X \mid \frac{w_1(c)}{w_2(c)} \sim R\}$.

Let $\Gamma = (G = (V, E, w), V_\Box, V_\Diamond)$ be a game on 2-dim VASS such that $W_\Box \neq \emptyset$, and let $v \in W_\Box$. We can define the following finite directed tree rooted at v. $T^{\Gamma,v} = (T_V^{\Gamma,v}, T_E^{\Gamma,v})$, where

$$\begin{aligned} T_V^{\Gamma,v} = \{p = (v = v_0, v_1, \ldots, v_k) \mid\ & p \text{ is a path in } G \wedge \\ & (\forall 0 \leq i < j < k)(v_i \neq v_j) \wedge \\ & (\forall 0 \leq i < k)(v_i \in W_\Box) \wedge \\ & (v_k \in W_\Diamond \vee (\exists 0 \leq i < k)(v_i = v_k))\ \} \end{aligned}$$

That is, the set of nodes of the tree is the set of paths in G starting from v and ending either at the first repeated vertex or at the first vertex from $\Diamond$'s winning region. If $p = (v_0, \ldots, v_k) \in T_V^{\Gamma,v}$, then $\mathsf{last}(p) = v_k$. We define depth of a node $p = (v_0, \ldots, v_k) \in T_V^{\Gamma,v}$ as $\mathsf{h}(p) = k$.

There is an edge $((v_0, \ldots, v_k), (u_0, \ldots, u_l)) \in T_E^{\Gamma,v}$ if and only if $l = k + 1$, for each $i \in \{0, \ldots, k\}$, $v_i = u_i$, and $(v_k, u_l) \in E$. If the game Γ is clear from the context, then the tree is denoted simply T^v. The set of nodes, T^v_V, is divided into inner nodes and leaves. The leaves of the tree are the nodes with no successors.

Let $q = (v_0, \ldots, v_k)$ be a leaf. If $\mathsf{last}(q) \notin W_\Diamond$, then $\mathsf{ce}(q) = (v_i, \ldots, v_k)$, $\mathsf{rh}(q) = i$, and $\mathsf{ph}(q) = (v_0, \ldots, v_i)$, where $i < k$ such that $v_i = v_k$. That is, $\mathsf{ce}(q)$ is the cycle closed at v_k, $\mathsf{rh}(q)$ is the depth of the node at which the closed cycle starts, and $\mathsf{ph}(q)$ is the path from the root to the starting vertex of the cycle. It holds that $\mathsf{rh}(q) = \mathsf{h}(\mathsf{ph}(q))$.

For the whole paper, let $\Gamma = (G = (V, E, w), V_\Box, V_\Diamond)$ be a game on 2-dim VASS. The elements of V will be called vertices. For each $v \in W_\Box$, the elements of T^v_V will be called nodes, inner nodes, leaves, or, when it is convenient, paths, because they are paths in G. We suppose that $|V| > 1$. For $|V| = 1$ the game is very simple to solve: There is only one vertex v with a self-loop, and so $v \in W_\Box$ if and only if the self-loop is in the set P.

3 The Proof

We prove that if $v \in W_\Box$, then $\Box$ has a strategy σ such that for each play $(v_0, v_1, v_2, \ldots)$ agreeing with σ, it holds that $(\forall k \in \mathbb{N}_0)(\sum_{i=0}^{k-1} w(v_i, v_{i+1}) \geq (K_{\min}, K_{\min}))$, where $K_{\min}$ is of polynomial size with respect to $|V|$. Therefore, we can reduce the problem of solving a game on 2-dim VASS to solving a reachability game with finite arena of polynomial size with respect to $|V|$, as described in the full version of this paper [2]. For reachability games there are polynomial-time algorithms. We first give an outline of the proof and then prove it formally.

3.1 Proof Outline

Each prefix $p^k_\infty = (v_0, \ldots, v_k)$ of an infinite path $p_\infty = (v_0, v_1, v_2, \ldots)$ in G can be partitioned using the following procedure: Start at v_0 and go along the path until an already visited vertex is encountered, then remove the closed cycle, leaving only the first vertex of the cycle, and continue in the same fashion. This way, p^k_∞ is partitioned into a set of cycles $c_1, \ldots, c_l$ and remaining path with no repeated vertex. If for each $i \in \{0, \ldots, k\}$, it holds that $v_i \in W_\Box$, then the partitioning corresponds to a traversal of the tree T^{v_0} in the following sense. The traversal starts at $(v_0) \in T^{v_0}_V$. When a leaf q is reached, $\mathsf{ce}(q)$ is added to the set of traversed cycles and the traversal continues at $\mathsf{ph}(q)$ until a node $p \in T^{v_0}_V$ such that $\mathsf{last}(p) = v_k$ is reached. The path p is the remaining path. The partitioning of paths into simple cycles plays a crucial role in our proof.

It is easy to see that if $\Box$ can ensure that only simple cycles from P are traversed, then he can win. However, this is not the only way he can win. $\Box$ can also win if he is able to balance the cycles from A and B. The cycles from A increase the first counter and decrease the second counter, and the cycles from B decrease the first counter and increase the the second counter. What is important are the ratios of the first and the second weights of the simple cycles. If $c_1 \in A$ and $c_2 \in B$ are the only simple cycles that can be traversed, and $\Box$ is able to alternate them arbitrarily, then he can win if and only if $\frac{w_1(c_1)}{w_2(c_1)} \leq \frac{w_1(c_2)}{w_2(c_2)}$, or, equivalently $w_1(c_1)w_2(c_2) \geq w_1(c_2)w_2(c_1)$. Moreover, he can alternate the cycles in such a way that both counters are always greater or equal to $-|V|$. Please note, that the set of all possible ratios of cycles in G from A and B is a subset of $\mathcal{R}$, and so it has at most $|V|^2$ elements.

If $v \in W_\Box$, then for each $R \in \mathcal{R}$, for a play starting at v, $\Box$ can ensure that only cycles from $A_{\leq R} \cup B_{\geq R} \cup P$ are traversed. This does not mean that $\Box$ can ensure that all these three types of cycles are traversed, we only claim that $\Box$ can ensure that each traversed cycle is from $A_{\leq R}$ or $B_{\geq R}$ or P. For example, consider the following situation winning for $\Box$. In this situation, $\Box$ can force only two cycles c_1 and c_2 such that $w(c_1) = (1, -1)$, $w(c_2) = (-1, 1)$, and these cycles have a common vertex so that $\Box$ is able to alternate between them. In this example, $\Box$ is not able to force a cycle from P and the ratio of both $c_1 \in A$ and $c_2 \in B$ is -1. Now, consider three cases: $R = -1$, $R < -1$, and $R > -1$. If $R = -1$, then $\Box$ is able to force a cycle from $A_{\leq R}$, namely, the cycle c_1, and he is also able to force a cycle from $B_{\geq R}$, namely, the cycle c_2. If $R < -1$, he is able to force a cycle only from $B_{\geq R}$, and if $R > -1$, then he is able to force a cycle only from $A_{\leq R}$. To sum up, in all the three cases, $\Box$ can ensure that only cycles from $A_{\leq R} \cup B_{\geq R} \cup P$ are traversed. To see why the claim holds in general, recall that each play in Γ starting at v corresponds to a traversal of the tree T^v.

Let $v \in W_\Box$ and $R \in \mathcal{R}$, then $\Box$ can ensure that all reached leaves in T^v correspond to cycles from $A_{\leq R} \cup B_{\geq R} \cup P$, because if $\Diamond$ could ensure that a leaf q such that $\mathsf{ce}(q) \in A_{>R} \cup B_{<R} \cup N$ is reached, then she would be able to ensure that *all* reached leaves correspond to a cycle from $\mathsf{ce}(q) \in A_{>R} \cup B_{<R} \cup N$ (Recall that if a leaf q is reached, then the play continues at $\mathsf{ph}(q)$). Therefore, if the play is long enough, then at least one of the counters goes below arbitrary constant. We omitted the possibility that a leaf q such that $\mathsf{last}(q) \in W_\Diamond$ is reached, because if such a leaf is reached, then $\Diamond$ can also win.

Unfortunately, the strategy of $\Box$, σ^v_R, that ensures that all traversed cycles are from $A_{\leq R} \cup B_{\geq R} \cup P$ may not be the sought strategy, because it may not be winning for $\Box$. The reason is that he may not be able to alternate the cycles from $A_{\leq R}$ and $B_{\geq R}$ so that both counters are always above some constant. For example, $\Diamond$ may be able to ensure that out of the cycles from $A_{\leq R} \cup B_{\geq R} \cup P$, only cycles from $A_{\leq R}$ are traversed, and so the second counter goes to $-\infty$. However, the sought strategy for $\Box$ can be assembled from the strategies σ^v_R for all $v \in W_\Box$ and $R \in \mathcal{R}$, albeit we may have to select much less number than $-|V|$ as the constant $K_{\min}$, but still polynomial with respect to $|V|$. The sought strategy is assembled in the following way.

Let $v \in W_\Box$ be the starting vertex. We select $R \in \mathcal{R}$ arbitrarily, and start using the strategy σ^v_R. We are using the strategy σ^v_R until there is a certain "disbalance" between the cycles from $A_{\leq R}$ and the cycles from $B_{\geq R}$. Let u be the current vertex when the disbalance occurs. If too many cycles from $A_{\leq R}$ were traversed, then we change the current strategy to $\sigma^u_{R'}$ such that the disbalance was caused by cycles from $A_{=R'}$ where $R' < R$, and if too many cycles from $B_{\geq R}$ were traversed, then we change the current strategy to $\sigma^u_{R''}$ such that the disbalance was caused by cycles from $B_{=R''}$ where $R'' > R$. The precise definition of the disbalance (it must be polynomial somehow) and the precise rules for selecting the new ratio will be given later. Before we get to the formal proof, one additional point has to be discussed.

From the previous paragraph, it follows that it is not enough that the strategy σ_R^v traverses only cycles from $A_{\leq R} \cup B_{\geq R} \cup P$. It must also be able to balance the cycles from $A_{=R}$ and $B_{=R}$, so that a disbalance between $A_{\leq R}$ and $B_{\geq R}$ is never caused by the cycles from $A_{=R}$ or $B_{=R}$. Therefore, the strategy σ_R^v we will define guarantees that only cycles from $A_{\leq R} \cup B_{\geq R} \cup P$ are traversed and the traversed cycles from $A_{=R}$ or $B_{=R}$ are kept in balance. The balance will not be the best possible as if we were able to alternate the cycles arbitrarily, but it will be such that the sum of the weights of the traversed cycles from $A_{=R} \cup B_{=R}$ is always greater or equal to $(-2 \cdot |V|^2, -2 \cdot |V|^2)$. This "balancing property" also ensures that if $R = \min \mathcal{R}$, then only the cycles from $B_{>R}$ can cause a disbalance, and if $R = \max \mathcal{R}$, then only the cycles from $A_{<R}$ can cause a disbalance.

3.2 Formal Proof

We will first define the "local" strategies for each $v \in W_\Box$ and $R \in \mathcal{R}$, and then we will assemble the "global" strategy from the local strategies. So let $v \in W_\Box$, $R \in \mathcal{R}$, and consider the tree T^v.

We define values of the nodes of the tree T^v: $\mathsf{value} : T_V^v \to \{-1, 0, \ldots, |V|\}^2 \cup \{\mathbf{0}, \mathbf{1}\}$.

The values are defined recursively. The values of leaves are defined as follows. Let $q = (v_0, \ldots, v_k) \in T_V^v$ be a leaf.

$$\mathsf{value}(p) = \begin{cases} \mathbf{0} & \text{if } \mathsf{last}(q) \in W_\Diamond \\ \mathbf{0} & \text{if } \mathsf{last}(q) \in W_\Box \wedge \mathsf{ce}(q) \in N \cup A_{>R} \cup B_{<R} \\ \mathbf{1} & \text{if } \mathsf{last}(q) \in W_\Box \wedge \mathsf{ce}(q) \in P \cup A_{<R} \cup B_{>R} \\ (\mathsf{rh}(q), -1) & \text{if } \mathsf{last}(q) \in W_\Box \wedge \mathsf{ce}(q) \in A_{=R} \\ (-1, \mathsf{rh}(q)) & \text{if } \mathsf{last}(q) \in W_\Box \wedge \mathsf{ce}(q) \in B_{=R} \end{cases} \quad (2)$$

To define the value of an inner node $p = (v_0, \ldots, v_k) \in T_V^v$, we introduce some notation:

$$\mathsf{amin}(p) = \min\{a \mid (\exists (p,q) \in T_E^v)(\mathsf{value}(q) = (a,b))\}$$

$$\mathsf{bmin}(p) = \min\{b \mid (\exists (p,q) \in T_E^v)(\mathsf{value}(q) = (a,b))\}$$

$$\mathsf{amax}(p) = \max\{a \mid (\exists (p,q) \in T_E^v)(\mathsf{value}(q) = (a,b))\}$$

$$\mathsf{bmax}(p) = \max\{b \mid (\exists (p,q) \in T_E^v)(\mathsf{value}(q) = (a,b))\}$$

If there is no successor of p with value from $\{-1, 0, \ldots, |V|\}^2$, i.e., all successors have the value $\mathbf{0}$ or $\mathbf{1}$, then $\mathsf{amin}(p) = \mathsf{bmin}(p) = \infty$ and $\mathsf{amax}(p) = \mathsf{bmax}(p) = -\infty$. If $\mathsf{last}(p) \in V_\Box$, then $\mathsf{value}(p)$ is defined as follows.

$$\mathsf{value}(p) = \begin{cases} \mathbf{0} & \text{if } (\forall (p,q) \in T^v_E)(\mathsf{value}(q) \neq \mathbf{1}) \wedge \\ & \quad (\mathsf{amin}(p) \geq \mathsf{h}(p) \vee \mathsf{bmin}(p) \geq \mathsf{h}(p)) \\ (\mathsf{amin}(p), \mathsf{bmin}(p)) & \text{if } (\forall (p,q) \in T^v_E)(\mathsf{value}(q) \neq \mathbf{1}) \wedge \\ & \quad \mathsf{amin}(p) < \mathsf{h}(p) \wedge \mathsf{bmin}(p) < \mathsf{h}(p) \\ \mathbf{1} & \text{if } (\exists (p,q) \in T^v_E)(\mathsf{value}(q) = \mathbf{1}) \end{cases} \tag{3}$$

If $\mathsf{last}(p) \in V_\Diamond$, then $\mathsf{value}(p)$ is defined as follows.

$$\mathsf{value}(p) = \begin{cases} \mathbf{1} & \text{if } (\forall (p,q) \in T^v_E)(\mathsf{value}(q) = \mathbf{1}) \\ (\mathsf{amax}(p), \mathsf{bmax}(p)) & \text{if } (\forall (p,q) \in T^v_E)(\mathsf{value}(q) \neq \mathbf{0}) \wedge \\ & \quad -\infty < \mathsf{amax}(p), \mathsf{bmax}(p) < \mathsf{h}(p) \\ \mathbf{0} & \text{if } (\exists (p,q) \in T^v_E)(\mathsf{value}(q) = \mathbf{0}) \vee \\ & \quad \mathsf{amax}(p) \geq \mathsf{h}(p) \vee \mathsf{bmax}(p) \geq \mathsf{h}(p) \end{cases} \tag{4}$$

The cycles from $N \cup A_{>R} \cup B_{<R}$ are called *bad cycles*. The cycles from $P \cup A_{<R} \cup B_{>R}$ are called *good cycles*. The cycles from $A_{=R} \cup B_{=R}$ are not given any special name.

We will show that the value of the root $(v) \in T^v_V$ is either $\mathbf{1}$ or $(-1, -1)$ (Please note that if $\mathsf{value}((v)) = (a, b)$, then the condition $a, b < \mathsf{h}((v)) = 0$ implies $a = b = -1$). We will show this by proving that if $\mathsf{value}((v)) = \mathbf{0}$, then $\Diamond$ has a winning strategy, which is in contradiction with $v \in W_\Box$. From the fact that the root has value $\mathbf{1}$ or $(-1, -1)$, we will infer a strategy for $\Box$ that ensures that only cycles from $A_{\leq R} \cup B_{\geq R} \cup P$ are traversed, and the cycles from $A_{=R}$ and $B_{=R}$ are kept in balance. So, let's first prove that the value of the root cannot be $\mathbf{0}$. We will only give a sketch of the proof, the whole formal proof is in the full version of this paper [2].

We will use a proof by contradiction. We will suppose that $\mathsf{value}((v)) = \mathbf{0}$ and show that $\Diamond$ has a strategy that ensures that for each $K \in \mathbb{Z}$, the first counter or the second counter will eventually go below K. The strategy is outlined below.

If $\mathsf{value}((v)) = \mathbf{0}$, then $\Diamond$ has a strategy that ensures that only cycles from $A_{\geq R} \cup B_{\leq R} \cup N$ are traversed or a leaf $q \in T^v_V$ such that $\mathsf{last}(q) \in W_\Diamond$ is reached. Moreover, she can choose the strategy in such a way that whenever a node p such that $\mathsf{value}(p) = \mathbf{0}$ is visited, then either the strategy ensures that if the next reached leaf r has $\mathsf{ce}(r) \in A_{=R} \cup B_{=R}$, then $\mathsf{ce}(r) \in A_{=R} \wedge \mathsf{rh}(r) \geq \mathsf{h}(p)$, or the strategy ensures that if the next reached leaf r has $\mathsf{ce}(r) \in A_{=R} \cup B_{=R}$, then $\mathsf{ce}(r) \in B_{=R} \wedge \mathsf{rh}(r) \geq \mathsf{h}(p)$. This allows $\Diamond$ to prevent $\Box$ from alternating cycles from $A_{=R}$ and $B_{=R}$. We note that $\Box$ may be able to perform a few alternations, because he can sometimes prevent $\Diamond$ from forcing the chosen kind of cycle, but only at the cost of visiting another node with value $\mathbf{0}$ that is deeper, and since the maximal depth is $|V|$, this cannot be repeated infinitely many times. Actually, $\Box$ may be able to perform infinite number of alternations, but at the cost of traversing a bad cycle infinitely many times.

To sum up, $\Diamond$ has a strategy that ensures that exactly one of the following four things happens. First, a leaf q such that $\mathsf{last}(q) \in W_\Diamond$ is reached. Second, only leaves q corresponding to bad cycles or cycles from $A_{=R} \cup B_{=R}$ are reached, and a bad cycle is traversed infinitely many times. Third, there is a point from which onwards all reached leaves q have $\mathsf{ce}(q) \in A_{=R}$. Fourth, there is a point

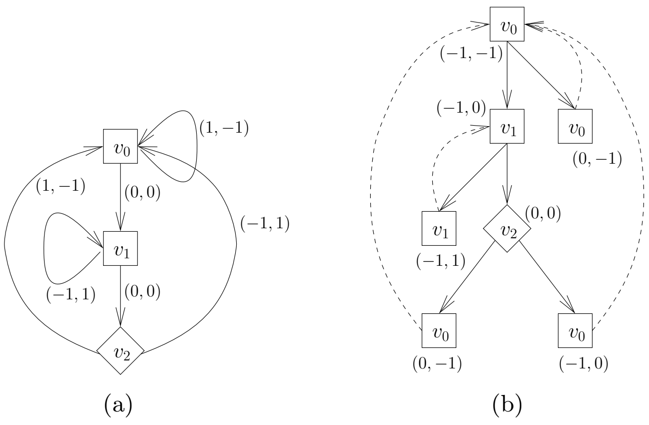

Fig. 1. Example tree valuation: (a) example game, (b) tree T^{v_0}

from which onwards all reached leaves q have $\mathsf{ce}(q) \in B_{=R}$. For the last three possibilities, at least one of the counters goes below arbitrary constant. For the first possibility, $\Diamond$ can apply her winning strategy from $\mathsf{last}(q) \in W_{\Diamond}$, and so at least one of the counters goes below arbitrary constant too. The whole formal proof is in the full version of this paper [2]. Therefore, there are only two possibilities for the value of the root: $\mathbf{1}$ and $(-1,-1)$. Let's now define the strategy for $\Box$ and show that it has the desired properties. We will start with some intuition.

The intuitive meaning of the node value $\mathbf{1}$ is that $\Box$ has a strategy to reach a leaf corresponding to a good cycle. The meaning of the node value (a,b) is more complex.

If a node p has the value (a,b), then $\Box$ has a strategy to reach a leaf corresponding to a good cycle or a cycle from $A_{=R} \cup B_{=R}$. Moreover, the strategy can be chosen in such a way that if the reached leaf q has $\mathsf{ce}(q) \in A_{=R}$, then $\mathsf{rh}(q) \le a$, or the strategy can be chosen in such a way that if the reached leaf q has $\mathsf{ce}(q) \in B_{=R}$, then $\mathsf{rh}(q) \le b$. In particular, if $a = -1$, then $\Box$ can force a good cycle or a cycle from $B_{=R}$ (albeit nothing can be said about the depth the play returns to), and if $b = -1$, then $\Box$ can force a good cycle or a cycle from $A_{=R}$ (albeit nothing can be said about the depth the play returns to). The rules for assigning values to nodes stipulate that $a < \mathsf{h}(p)$ and $b < \mathsf{h}(p)$. This is important for balancing the cycles from $A_{=R}$ and $B_{=R}$.

The player $\Box$ may not be able to alternate the cycles from $A_{=R}$ and $B_{=R}$ arbitrarily, as Figure 1 shows. In Figure 1 (a), there is a game on 2-dim VASS. Squares are $\Box$'s vertices and the diamond is a $\Diamond$'s vertex. The pairs of numbers are weights of the edges depicted as arrows. In this game, $\Box$ can win from all vertices. Let $R = -1$, then all cycles in the figure are from $A_{=R} \cup B_{=R}$. In Figure 1 (b), there is the tree T^{v_0}. The pairs of numbers are values of the nodes and the dashed arrows emanating from leaves show, for each leaf, where the game (projected on the tree) continues when it reaches the leaf.

If the play starts from v_0, then at the beginning, $\Box$ is able to traverse a cycle from $A_{=R}$ arbitrary number of times (the cycle (v_0, v_0)), after that he is able to traverse a cycle from $B_{=R}$ arbitrary number of times (the cycle (v_1, v_1)). However, after that he is not able to start traversing cycles from $A_{=R}$ immediately. The value of the node $(v_0, v_1) \in T_V^{v_0}$ at depth 1 is $(-1, 0)$, which indicates that $\Box$ is able to force a cycle from $B_{=R}$, but not from $A_{=R}$. However, $\Box$ has a strategy that ensures that if $\Diamond$ forces a cycle from $B_{=R}$, then the play returns to a node at smaller depth, namely, the depth 0. At the depth 0, $\Box$ is, again, able to force a cycle from $A_{=R}$.

In general, we claim that $\Box$ has a strategy that ensures that only good cycles and cycles from $A_{=R} \cup B_{=R}$ are traversed. Moreover, the strategy also ensures that both the sum of the first weights of the cycles from $A_{=R} \cup B_{=R}$ and the sum of the second weights of the cycles from $A_{=R} \cup B_{=R}$ is always greater or equal to $-2 \cdot |V|^2$.

In the case where $\mathsf{value}((v)) = \mathbf{1}$, $\Box$ has a strategy to traverse only the good cycles and the claim obviously holds.

The second case is that $\mathsf{value}((v)) = (-1, -1)$. In this case, $\Box$ has a strategy that ensures that only nodes p with value $\mathbf{1}$ or value $(a, b) \in \{-1, \ldots, |V|\}^2$ are visited. This alone implies that only good cycles and cycles from $A_{=R} \cup B_{=R}$ are traversed. Moreover, he is able to choose the strategy in such a way that it balances the cycles from $A_{=R}$ and $B_{=R}$. When a disbalance between the cycles from $A_{=R}$ and $B_{=R}$ occurs, let's say that too many cycles from $A_{=R}$ have been traversed, then $\Box$ aims to traverse cycles from $B_{=R}$ or good cycles. If the current node p has the value $\mathbf{1}$, then $\Box$ can ensure that the next traversed cycle is a good cycle, which does not worsen the disbalance. If p has the value (a, b) and $a \neq -1$, $\Diamond$ can force a cycle from $A_{=R}$, but if she does, $\Box$ can ensure that the play returns to the depth a or smaller, and since $a < \mathsf{h}(p)$, we return to a smaller depth than the depth of p. We can continue using the same reasoning and conclude that after traversing at most $|V| - 1$ (maximal depth of an inner tree node) "unwanted" cycles from $A_{=R}$, we get to the root, where $\Box$ can force cycles from $B_{=R}$ or good cycles. Therefore, he can alleviate the disbalance caused by the cycles from $A_{=R}$. Of course, the case when a disbalance is caused by cycles from $B_{=R}$ is symmetric. Formal definition of this "balancing" strategy of $\Box$ is as follows.

A general strategy of $\Box$ is a function $\sigma : V^* \cdot V \to V$, i.e., it decides based on the whole history of the play. However, the strategy we define now will decide only based on the current node of the tree T_V^v (which consists of fragments of the whole history) and some additional memory which could be computed from the complete history of the play.

Apart from the current node, the player $\Box$ keeps a triple $(x, y, z) \in \{-2 \cdot |V|^2, \ldots, 0, \ldots, 2 \cdot |V|^2\}^2 \times \{0, 1\}$, where x is the sum of the first weights of the traversed cycles from $A_{=R} \cup B_{=R}$, y is the sum of the second weights of the traversed cycles from $A_{=R} \cup B_{=R}$, and z is the mode of the strategy: $z = 0$ means that the strategy aims to traverse cycles from $A_{=R}$, and $z = 1$ means that the strategy aims to traverse cycles from $B_{=R}$. The memory plays a crucial role in keeping the traversed cycles from $A_{=R}$ and $B_{=R}$ in balance.

The strategy of $\Box$ visits only nodes p of the tree T^v with $\mathsf{value}(p) = \mathbf{1}$ or $\mathsf{value}(p) = (a,b)$ (recall that $a, b < \mathsf{h}(p)$). The play starts at the root (v). It holds that $\mathsf{h}((v)) = 0$, and $\mathsf{value}((v)) = \mathbf{1}$ or $\mathsf{value}((v)) = (-1,-1)$. Initial state of the memory is $(0,0,0)$. Let's consider a general situation where we are at the inner node p such that $\mathsf{last}(p) \in V_\Box$, $\mathsf{value}(p) = \mathbf{1}$ or $\mathsf{value}(p) = (a,b)$ such that $a, b < \mathsf{h}(p)$, and the current state of memory is (x,y,z), then the strategy of $\Box$, denoted by σ, works as follows. Please note that the strategy does not have to consider leaves, because at each leaf q, the play automatically returns to the inner node $\mathsf{ph}(q)$. First, how a successor is chosen:

$$\sigma(p,(x,y,z)) = \begin{cases} q \text{ if } (p,q) \in T_E \wedge \mathsf{value}(p) = \mathbf{1} \wedge \mathsf{value}(q) = \mathbf{1} \\ q \text{ if } (p,q) \in T_E \wedge \mathsf{value}(p) = (a,b) \wedge z = 0 \wedge \mathsf{value}(q) = (a',b) \\ q \text{ if } (p,q) \in T_E \wedge \mathsf{value}(p) = (a,b) \wedge z = 1 \wedge \mathsf{value}(q) = (a,b') \end{cases} \tag{5}$$

Please note that for a node p with $\mathsf{value}(p) = (a,b)$, the existence of a successor with value (a',b) and the existence of a successor with value (a,b') follows from (3). It is also possible that these are not two distinct successors but only one with value (a,b).

Second, how the memory is updated. The memory is updated only when a leaf is reached, so let's suppose that we have reached the leaf q. Then the memory (x,y,z) is updated to:

$$\begin{array}{ll}
(x,y,z) & \text{if } \mathsf{ce}(q) \text{ is a good cycle} \\
(x+w_1(\mathsf{ce}(q)), y+w_2(\mathsf{ce}(q)), z) & \text{if } \mathsf{ce}(q) \in A_{=R} \cup B_{=R} \wedge \\
& \quad z = 0 \wedge \\
& \quad (\, x+w_1(\mathsf{ce}(q)) < 0 \vee \\
& \quad\quad (\, x+w_1(\mathsf{ce}(q)) \in [-|V|^2, |V|^2] \wedge \\
& \quad\quad\quad y+w_2(\mathsf{ce}(q)) \in [-|V|^2, |V|^2] \,)\,) \\
(x+w_1(\mathsf{ce}(q)), y+w_2(\mathsf{ce}(q)), z) & \text{if } \mathsf{ce}(q) \in A_{=R} \cup B_{=R} \wedge \\
& \quad z = 1 \wedge \\
& \quad (\, y+w_2(\mathsf{ce}(q)) < 0 \vee \\
& \quad\quad (\, x+w_1(\mathsf{ce}(q)) \in [-|V|^2, |V|^2] \wedge \\
& \quad\quad\quad y+w_2(\mathsf{ce}(q)) \in [-|V|^2, |V|^2] \,)\,) \\
(x+w_1(\mathsf{ce}(q)), y+w_2(\mathsf{ce}(q)), 1) & \text{if } \mathsf{ce}(q) \in A_{=R} \cup B_{=R} \wedge \\
& \quad z = 0 \wedge \\
& \quad x+w_1(\mathsf{ce}(q)) \geq 0 \wedge \\
& \quad (\, |x+w_1(\mathsf{ce}(q))| > |V|^2 \vee \\
& \quad\quad |y+w_2(\mathsf{ce}(q))| > |V|^2 \,) \\
(x+w_1(\mathsf{ce}(q)), y+w_2(\mathsf{ce}(q)), 0) & \text{if } \mathsf{ce}(q) \in A_{=R} \cup B_{=R} \wedge \\
& \quad z = 1 \wedge \\
& \quad y+w_2(\mathsf{ce}(q)) \geq 0 \wedge \\
& \quad (\, |x+w_1(\mathsf{ce}(q))| > |V|^2 \vee \\
& \quad\quad |y+w_2(\mathsf{ce}(q))| > |V|^2 \,)
\end{array} \tag{6}$$

We note again that if a leaf q is reached, then the play automatically continues at node $\mathsf{ph}(q)$, and so the play is infinite. Let's now take a closer look at the memory updates.

While $x, y \in [-|V|^2, |V|^2]$, the strategy does not change the type of cycles it aims for, z is not changed (first 3 items in (6)). When $|x|$ or $|y|$ exceeds $|V|^2$, $z = 0$, and $x \geq 0$, it means that too many cycles from $A_{=R}$ have been traversed. Therefore z is changed to 1, and so the strategy aims for cycles from $B_{=R}$ (4th item in (6)). As described before, even after this action, some cycles from $A_{=R}$ may be traversed before cycles from $B_{=R}$, but there can be at most $|V| - 1$ of these unwanted cycles, therefore x and y do not leave the interval $[-2 \cdot |V|^2, 2 \cdot |V|^2]$. The situation where too many cycles from $B_{=R}$ have been traversed is dealt with analogously (5th item in (6)).

The following two lemmas show that the strategy σ satisfies the desired properties. An intuition why they hold was already given. Their formal proofs are in the full version of this paper [2].

Lemma 1. *Let $\Gamma = (G = (V, E, w), V_\Box, V_\Diamond)$ be a game on 2-dim VASS. Let further $v \in W_\Box$ be the starting vertex, $R \in \mathcal{R}$, and let the strategy σ be defined as in (5). Then the following holds. If the value of the root (v) of the tree T^v is $\mathsf{value}((v)) = \mathbf{1}$, then the strategy σ ensures that only nodes p with $\mathsf{value}(p) = \mathbf{1}$ are visited. If the value of the root (v) of the tree T^v is $\mathsf{value}((v)) = (-1, -1)$, then the strategy σ ensures that only nodes p with $\mathsf{value}(p) = \mathbf{1}$ or $\mathsf{value}(p) = (a, b)$ such that $a, b < \mathsf{h}(p)$ are visited.* ■

Lemma 1 implies that only good cycles and cycles from $A_{=R} \cup B_{=R}$ are traversed. The next lemma states that the cycles from $A_{=R}$ and $B_{=R}$ are kept in balance.

Lemma 2. *Let $\Gamma = (G = (V, E, w), V_\Box, V_\Diamond)$ be a game on 2-dim VASS. Let further $v \in W_\Box$ be the starting vertex, $R \in \mathcal{R}$, let the root (v) of the tree T^v have the value $\mathbf{1}$ or $(-1, -1)$, and let the strategy σ be defined as in (5). Let $\Box$ use the strategy σ, let $\Diamond$ use arbitrary strategy. The outcome of these two strategies corresponds to a sequence of nodes $(p_0, p_1, p_2, \ldots)$. Let $(q_0, q_1, q_2, \ldots)$ be the subsequence of the sequence containing* all *reached leaves corresponding to cycles from $A_{=R} \cup B_{=R}$. In particular, for each $i \in \mathbb{N}_0$, $\mathsf{ce}(q_i) \in A_{=R} \cup B_{=R}$. Then for each $k \in \mathbb{N}_0$, it holds that $|\sum_{i=0}^{k} w_j(\mathsf{ce}(q_i))| \leq 2 \cdot |V|^2$ where $j = 0, 1$.* ■

For technical convenience, let's number the elements of the set of the cycle ratios, namely, let $\mathcal{R} = \{R_1, \ldots, R_{|\mathcal{R}|}\}$ where $R_1 < \cdots < R_{|\mathcal{R}|}$. It holds that $|\mathcal{R}| \leq |V|^2$. By Lemma 2, for each $v \in W_\Box$ and R_k such that $k \in \{1, \ldots, |\mathcal{R}|\}$, the player $\Box$ has the strategy σ_k^v that ensures that only cycles from $P \cup A_{\leq R_k} \cup B_{\geq R_k}$ are traversed. Moreover, the cycles from $A_{=R_k}$ and $B_{=R_k}$ are balanced in the sense that the absolute value of both components of the sum of their weights never exceeds $2 \cdot |V|^2$. Also, when using the strategy σ_k^v, the play never leaves the set $W_\Box$.

Using the above facts, we will now assemble a global strategy σ of $\Box$ such that there is a constant $K_{\min} \in \mathbb{Z}$ of polynomial size with respect to $|V|$ such that whatever strategy π the opponent $\Diamond$ uses, the resulting infinite play $\mathsf{outcome}^{\Gamma}(v_0, \sigma, \pi) = (v_0, v_1, v_2, \ldots)$ satisfies the following. For each $k \in \mathbb{N}_0$, $\sum_{i=0}^{k-1} w_1(v_i, v_{i+1}) \geq K_{\min}$ and $\sum_{i=0}^{k-1} w_2(v_i, v_{i+1}) \geq K_{\min}$. The strategy σ will be assembled from the strategies σ_k^v where $v \in W_\Box$ and $k \in \{1, \ldots, |\mathcal{R}|\}$. So, let's describe how this is done.

Each strategy σ_k^v has the three-component memory as described before. Let $k \in \{1, \ldots, |\mathcal{R}|\}$. For each $v \in W_\Box$, the strategy σ_k^v balances the cycles from $A_{=R_k}$ and $B_{=R_k}$. We will let all the strategies with the same k use the same three-component memory. Therefore, the global strategy σ will have $|\mathcal{R}|$ three-component memories, one for each k. For a specific $k \in \{1, \ldots, |\mathcal{R}|\}$, the tree-component memory will be denoted by (x_k, y_k, z_k). Apart from that, σ will have additional memory that consists of two $|\mathcal{R}|$-tuples. The first $|\mathcal{R}|$-tuple will be $(a_1, \ldots, a_{|\mathcal{R}|}) \in \{0, \ldots, 4 \cdot |V|^4 + 3 \cdot |V|\}^{|\mathcal{R}|}$ and it will store the sums of the first weights of the traversed cycles from A, separately for each ratio. The second $|\mathcal{R}|$-tuple will be $(a'_1, \ldots, a'_{|\mathcal{R}|}) \in \{-4 \cdot |V|^4 - |V|, \ldots, 0\}^{|\mathcal{R}|}$ and it will store the sums of the first weights of the traversed cycles from B, separately for each ratio. However, when using the strategy σ_k^v, only traversed cycles from A and B with ratios R_i such that $i \neq k$ will be recorded in the additional memory. The traversed cycles with the ratio R_k will be recorded only in the three-component memory (x_k, y_k, z_k). The global strategy σ will also remember which strategy σ_k^v it is currently using by remembering the vertex v and the integer k. The strategy σ is defined as follows.

We will not describe (again) how the three-component memories are used and updated, we will only describe how the two additional $|\mathcal{R}|$-tuples are handled. The current tree that the strategy is working with is denoted by $T^v = (T_V^v, T_E^v)$ where (v) is the root of the tree. Let v, k, $(a_1, \ldots, a_{|\mathcal{R}|})$, $(a'_1, \ldots, a'_{|\mathcal{R}|})$ be the current state of the additional memory, and let p be the current inner node in the current tree T^v. We will first describe how the strategy decides and then how the memory is updated. The strategy decides as follows:

$$\sigma(p, v, k, (a_1, \ldots, a_{|\mathcal{R}|}), (a'_1, \ldots, a'_{|\mathcal{R}|})) = \sigma_k^v(p) \tag{7}$$

Now, let us describe how the memory is updated. The initial state of the memory is $(v, 1, (0, \ldots, 0), (0, \ldots, 0))$ where v is the vertex the play starts from, and so the first tree the strategy σ works with is the tree T^v rooted at v, and the first used substrategy is σ_1^v. The two $|\mathcal{R}|$-tuples in the memory play a crucial role in keeping the traversed cycles from A and B in balance. As was already mentioned, the first $|\mathcal{R}|$-tuple records the sums of the first weights of the traversed cycles from A. There are two bounds that bound the elements of the tuple from above: a soft bound and a hard bound. The soft bound is equal to $4 \cdot |V|^4 + 2 \cdot |V|$ and we denote it by C_A. If some element a_i exceeds the soft bound, then the strategy takes some actions so that a_i is not increased further and it never exceeds the hard bound $\bar{C}_A$ which is equal to $4 \cdot |V|^4 + 3 \cdot |V|$. Similarly, there is a soft bound and a hard bound for the second $|\mathcal{R}|$-tuple. The second $|\mathcal{R}|$-tuple records the sums of the first weights of the traversed cycles from B. Unlike for the first tuple, the bounds for the second tuple bound the elements of the tuple from below. The soft bound is $C_B = -4 \cdot |V|^4$, and the hard bound is $\bar{C}_B = -4 \cdot |V|^4 - |V|$. Before explaining the actions the strategy takes to ensure that the hard bounds are never exceeded, we describe precisely how the memory is updated. It is updated only when a leaf in the current tree is reached, so let's suppose that we have reached the leaf q. Then the memory $(v, k, (a_1, \ldots, a_{|\mathcal{R}|}), (a'_1, \ldots, a'_{|\mathcal{R}|}))$ is updated to:

$$\begin{array}{ll}
(v, k,\ (a_1, \ldots, a_{|\mathcal{R}|}), & \text{if } \mathsf{ce}(q) \in P \cup A_{=R_k} \cup B_{=R_k} \\
\qquad (a'_1, \ldots, a'_{|\mathcal{R}|})) & \\
(v, k,\ (a_1, \ldots, a_i + w_1(\mathsf{ce}(q)), \ldots, a_{|\mathcal{R}|}), & \text{if } \mathsf{ce}(q) \in A_{=R_i} \wedge \\
\qquad (a'_1, \ldots, a'_{|\mathcal{R}|})) & \quad i < k \wedge \\
 & \quad a_i + w_1(\mathsf{ce}(q)) \leq C_A \\
(v, k,\ (a_1, \ldots, a_{|\mathcal{R}|}), & \text{if } \mathsf{ce}(q) \in B_{=R_j} \wedge \\
\qquad (a'_1, \ldots, a'_j + w_1(\mathsf{ce}(q)), \ldots, a'_{|\mathcal{R}|})) & \quad j > k \wedge \\
 & \quad a'_j + w_1(\mathsf{ce}(q)) \geq C_B \\
(v, k,\ (a_1, \ldots, a_i - a_i, \ldots, a_{|\mathcal{R}|}), & \text{if } \mathsf{ce}(q) \in A_{=R_i} \wedge \\
\qquad (a'_1, \ldots, a'_j - a'_j, \ldots, a'_{|\mathcal{R}|}) & \quad i < k \wedge \\
 & \quad a_i + w_1(\mathsf{ce}(q)) > C_A \wedge \\
 & \quad j > i \wedge a'_j < C_B \\
(v, k,\ (a_1, \ldots, a_i - a_i, \ldots, a_{|\mathcal{R}|}), & \text{if } \mathsf{ce}(q) \in B_{=R_j} \wedge \\
\qquad (a'_1, \ldots, a'_j - a'_j, \ldots, a'_{|\mathcal{R}|}) & \quad j > k \wedge \\
 & \quad a'_j + w_1(\mathsf{ce}(q)) < C_B \wedge \\
 & \quad i < j \wedge a_i > C_A \\
(\mathsf{last}(q), i,\ (a_1, \ldots, a_i + w_1(\mathsf{ce}(q)), \ldots, a_{|\mathcal{R}|}), & \text{if } \mathsf{ce}(q) \in A_{=R_i} \wedge \\
\qquad (a'_1, \ldots, a'_{|\mathcal{R}|})) & \quad i < k \wedge \\
 & \quad a_i + w_1(\mathsf{ce}(q)) > C_A \wedge \\
 & \quad (\nexists j > i)(a'_j < C_B) \\
(\mathsf{last}(q), j,\ (a_1, \ldots, a_{|\mathcal{R}|}), & \text{if } \mathsf{ce}(q) \in B_{=R_j} \wedge \\
\qquad (a_1, \ldots, a'_j + w_1(\mathsf{ce}(q)), \ldots, a'_{|\mathcal{R}|})) & \quad j > k \wedge \\
 & \quad a'_j + w_1(\mathsf{ce}(q)) < C_B \wedge \\
 & \quad (\nexists i < j)(a_i > C_A)
\end{array} \tag{8}$$

We note that if a leaf q is reached, then there are two possibilities as to which node the play continues at. The first case is when σ does not change the substrategy σ^v_k (first 5 items in (8)). In this case the play continues at node $\mathsf{ph}(q)$. The second case is when σ does change the substrategy σ^v_k (last 2 items in (8)). In this case the play continues at the root ($\mathsf{last}(q)$) of the new tree $T^{\mathsf{last}(q)}$.

Before getting to formal proofs we will describe how the definition of the strategy σ corresponds to what was said in Section 3.1.

While using the substrategy σ^v_k, only cycles from $P \cup A_{\leq R_k} \cup B_{\geq R_k}$ are traversed. Moreover, by Lemma 2, the effects of the cycles from $A_{=R_k}$ and $B_{=R_k}$ are balanced. The additional memory of the global strategy σ is used to detect a disbalance between the cycles from $A_{<R_k}$ and $B_{>R_k}$. A disbalance is suspected when some a_i such that $i < k$ goes above $C_A = 4 \cdot |V|^4 + 2 \cdot |V|$, or some a'_j such that $j > k$ goes below $C_B = -4 \cdot |V|^4$. However, this does not necessarily imply a disbalance. We will look only at the first case, the other one is symmetric. If some a_i such that $i < k$ goes above C_A, and there is also some $j > i$ such that a'_j is below C_B, then there is no disbalance. The effects of the corresponding cycles from $A_{=R_i}$ and $B_{=R_j}$ balance each other. The bounds were selected so that the sum of the weights of these cycles is greater or equal to $(|V|, |V|)$, which justifies the zeroing of the appropriate elements of the memory (4th item in (8)) and also compensates for the possibly negative simple paths that are “lost” when switching a substrategy.

A substrategy is changed when there is no a'_j that would compensate for a_i, and so a disbalance occurs. The substrategy is changed to σ_i^u where u is the current vertex when the disbalance occurred (6th item in (8)). It holds that $u = \mathsf{last}(q)$ where q is the appropriate leaf in the tree T^v, the visit of which caused the disbalance. The substrategy is changed to ensure that a_i is not further increased. The substrategy σ_i^u works with the tree T^u, and so the path $\mathsf{ph}(q)$ from v to u is lost in the sense that it is reflected neither in the local nor in the global memory. However, as mentioned above this lost paths are compensated for, and so the global memory together with the local memories gives a lower bound on the first counter, and indirectly also on the second counter. Since all the components of the memories are of polynomial size, so are the lower bounds.

The following theorem makes the above arguments precise. Its formal proof is in the full version of this paper [2].

Theorem 1. *Let $\Gamma = (G = (V, E, w), V_\Box, V_\Diamond)$ be a game on 2-dim VASS. Let further $v \in W_\Box$ be the starting vertex, and let $\Box$ use the strategy σ as defined in (7). Let $\Diamond$ use an arbitrary strategy π, and let $\mathsf{outcome}^\Gamma(v, \sigma, \pi) = (v = v_0, v_1, v_2, \ldots)$ be the resulting play. Let k be the state of the play after k steps, i.e., we are at the vertex v_k. Let $(v', l, (a_1, \ldots, a_{|\mathcal{R}|}), (a'_1, \ldots, a'_{|\mathcal{R}|}))$ be the state of the global memory, and let $(x_i, y_i, z_i)_{i\in\{1,\ldots,|\mathcal{R}|\}}$ be the state of the local memories of the substrategies. Then the following holds:*

$$\textstyle\sum_{i=0}^{k-1} w_1(v_i, v_{i+1}) \geq \sum_{i\in\{1,\ldots,|\mathcal{R}|\}}(a_i + a'_i + x_i) - |V|^3 - 2\cdot|V|$$
$$\textstyle\sum_{i=0}^{k-1} w_2(v_i, v_{i+1}) \geq \sum_{i\in\{1,\ldots,|\mathcal{R}|\}}(-|V|a_i - \frac{1}{|V|}a'_i + y_i) - |V|^3 - 2\cdot|V| \quad \blacksquare$$

For each $i \in \{1, \ldots, |\mathcal{R}|\}$, it holds that $0 \leq a_i \leq \bar{C}_A = 4\cdot|V|^4 + 3\cdot|V|$, and $-4\cdot|V|^4 - |V| = \bar{C}_B \leq a'_i \leq 0$, and $x_i, y_i \in [-2\cdot|V|^2, 2\cdot|V|^2]$. Therefore, by Theorem 1, if we set $K_{\min}$ to, for example, $-100\cdot|V|^7$, then for each play $(v_0, v_1, v_2, \ldots)$ with $v_0 \in W_\Box$, agreeing with the strategy σ, it holds that $(\forall k \in \mathbb{N}_0)(\sum_{i=0}^{k-1} w(v_i, v_{i+1}) \geq (K_{\min}, K_{\min}))$.

Acknowledgement. I would like to thank to Tomáš Brázdil, Petr Jančar, and Antonín Kučera for fruitful consultations and advice.

References

1. Brázdil, T., Jančar, P., Kučera, A.: Reachability games on extended vector addition systems with states. In: Proc. International Colloquium on Automata, Languages and Programming. LNCS. Springer, Heidelberg (2010)
2. Chaloupka, J.: Z-reachability problem for games on 2-dimensional vector addition systems with states is in P. Technical Report FIMU-RS-2010-06, Faculty of Informatics, Masaryk University, Brno, Czech Republic (2010)

3. Hopcroft, J.E., Pansiot, J.-J.: On the reachablility problem for 5-dimensional vector addition systems. Theoretical Computer Science 8(2), 135–159 (1979)
4. Jančar, P.: Undecidability of bisimilarity for petri nets and related problems. Theoretical Computer Science 148, 281–301 (1995)
5. Jančar, P.: Decidability of bisimilarity for one-counter processes. Information and Computation 158, 1–17 (2000)
6. Reisig, W.: Petri Nets – An Introduction. Springer, Heidelberg (1985)