Reachability games and related matrix and word problems

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Reino Niskanen

February 2018

In memory of my brother

Arvo Niskanen

1985–2016

Abstract

In this thesis, we study different two-player zero-sum games, where one player, called Eve, has a reachability objective (i.e., aims to reach a particular configuration) and the other, called Adam, has a safety objective (i.e., aims to avoid the configuration). We study a general class of games, called Attacker-Defender games, where the computational environment can vary from as simple as the integer line to n-dimensional topological braids. Similarly, the moves themselves can be simple vector addition or linear transformations defined by matrices. The main computational problem is to decide whether Eve has a winning strategy to reach the target configuration from the initial configuration, or whether the dual holds, that is, whether Adam can ensure that the target is never reached. The notion of a winning strategy is widely used in game semantics and its existence means that the player can ensure that his or her winning conditions are met, regardless of the actions of the opponent. It general, games provide a powerful framework to model and analyse interactive processes with uncontrollable adversaries.

We formulated several Attacker-Defender games played on different mathematical domains with different transformations (moves), and identified classes of games, where the checking for existence of a winning strategy is undecidable. In other classes, where the problem is decidable, we established their computational complexity. In the thesis, we investigate four classes of games where determining the winner is undecidable: *word games*, where the players' moves are words over a group alphabet together with integer weights or where the moves are pairs of words over group alphabets; *matrix games on vectors*, where players transform a three-dimensional vector by linear transformations defined by 3×3 integer matrices; *braid games*, where players braid and unbraid a given braid; and last, but not least, games played on *two-dimensional $\mathbb{Z}$-VAS*, closing the gap between decidable and undecidable cases and answering an existing open problem of the field. We also identified decidable fragments, such as *word games*, where the moves are over a single group alphabet, games on *one-dimensional $\mathbb{Z}$-VASS*. For word games, we provide an upper-bound of EXPTIME, while for games on $\mathbb{Z}$-VASS, tight bounds of EXPTIME-complete or EXPSPACE-complete, depending on the state structure. We also investigate single-player systems such as polynomial iteration and identity problem in matrix semigroups. We show that the reachability problem for polynomial iteration is PSPACE-complete while the identity problem for the Heisenberg group is in PTIME for dimension three and in EXPTIME for higher dimensions.

Acknowledgements

First and foremost, I would like to thank my supervisor, Prof. Igor Potapov, for his guidance and support from day one. Without his insights, helpful comments and constructive criticism, this thesis would not be possible.

I would also like to thank my second supervisor, Dr Vesa Halava, for his unique combination of laidback attitude and scientific rigour, not to mention sense of humour, and my third supervisor, Prof. Paul Spirakis, for his useful advice.

I'm grateful to Prof. Sven Schewe and Dr Dietmar Berwanger for agreeing to act as examiners of my thesis and for the time they spent on reviewing it.

The Department of Computer Science of University of Liverpool provided excellent facilities for doing a PhD. A special thank you to fellow PhD students, academic and support staff of the department. Especially, my office mates, Tom, Ashley, Sang-Ki and Pavel, who have made the last three years into a very enjoyable experience. I would also like to thank the Department of Computer Science for the scholarship that made my studies possible. I also thank Nokia Foundation for the scholarship.

My gratitude is extended to my co-authors, Prof. Tero Harju, Dr Sang-Ki Ko and Dr Julien Reichert, for their input, and helping me turn my vague ideas into concrete statements and proofs. Collaboration with experienced co-authors helped me to produce more interesting results and also to shape my thesis in a better way.

Obviously, I'm indebted to my family. To my parents, Irina and Leo, for setting me on this path and helping me on every step of the way. To my siblings, Lauri, Arvo, Leo and Emma, and to Ildiko, their support has been invaluable. Last but not least, to Ulla, for bearing with trials and tribulations of my everyday life.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Games are encountered in almost every aspect of interactions of living creatures. Kids learn about the surrounding world through different games, first when interacting with their parents or just the environment, then with other children and adults. Recreational games, such as tic-tac-toe, chess, go or video games, allow us to better ourselves. Often the origins of games are in the real world. Game such as chess or battleships are abstraction of warfare, while Monopoly is an abstraction of economy and wealth creation.

In the modern world, the reliability of a software code and verification of the correct functionality of complex technological devices requires the analysis of various interactive processes and open systems, where it is important to take into account the effects of uncontrollable adversaries, such as environment or malicious users. Two-player computational games provide a powerful framework for problems related to verification and refinement of reactive systems [8], and have deep connections with automata theory and logic [103, 146]. Infinite-state games can be classified according to the winning conditions, such as parity [3], energy [67], counter reachability, or a combination of two or more winning conditions [38]. The extensions of classical reachability problems to game schemes, studied in different contexts and settings, have recently garnered considerable interest [2,3,27,29,38,67,136,138]. Such games provide a powerful mathematical framework for a large number of computational problems.

One of the main computational problem for games is to check whether one of the players can win the game regardless of what the opponent does. In other words, whether there exists a winning strategy in the game. In many cases of high-dimensional games, the problem of checking for the existence of a winning strategy can be computationally hard

and even undecidable. Answering the same question for low-dimensional systems can also be a challenging problem. This is either due to a lack of tools for the analysis of complex dynamics or due to a lack of "space" to encode directly the universal computations to show that the problem is undecidable.

In this thesis, we study two-player turn-based zero-sum games with perfect information that we call *Attacker-Defender games*. An Attacker-Defender game is played in rounds, where in each round a move of Defender is followed by a move of Attacker starting from some initial position. There is no randomness or hidden information in the game, that is, both players are aware of all information about gameplay, such as possible moves of their opponent. The aim of Attacker is to reach a target position while Defender tries to keep Attacker from reaching the target position. Then, we say that Attacker has a winning strategy if she can eventually reach the target position regardless of Defender's moves. The main computational problem we study is to decide which of the players wins based on a given set of eligible moves, computational environment and reachability objectives.

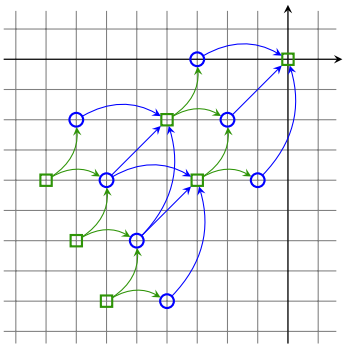
We show that in a number of restricted cases of such games, it is not possible to decide whether a winning strategy exists for a given set of moves, an initial position and a target position. On the other hand, for some other games, we provide tight complexity bounds for the decision problem.

As a side note, in the literature, there are no fixed names for the players. The player with the existential objective (reachability in our games) is often called Player 1, System, Model, Attacker, Eve (sometimes written as ∃ve), while the player with the universal objective (safety in our games) is called Player 2, Environment, Specification, Defender, Adam (sometimes written as ∀dam). In this thesis, we use the latter pair, that is, Eve and Adam. This might be a bit confusing as the general game framework is still called Attacker-Defender game as that was the terminology we used during the initial research.

Following early results for games on VASS (Vector Addition Systems with States)[1] [1, 2, 29], Doyen and Rabinovich formulated an open problem about the simplest version of games (*robot games*) for which the decidability was unknown [62]. We call robot games $\mathbb{Z}$-VAS games, as in our eyes, the game can be seen as a game played on integer vector addition system and has little to do with robots[2]. That is, $\mathbb{Z}$-VAS games are two-player games played by updating a vector of d integer counters. Each of the players, Adam and Eve, has a finite set of vectors in $\mathbb{Z}^d$. As in general Attacker-Defender games, a play starts

[1] A game is played on a graph with states of Eve and states of Adam, with $\mathbb{N}^2$ as a vector space.
[2] A fact pointed out by several reviewers.

from a given initial vector $\mathbf{x}_0 \in \mathbb{Z}^d$, and proceeds in rounds. During each round, first Adam adds a vector from his set, followed by Eve doing the same. Eve wins when, after her turn, the vector is the zero vector. A simple example of a two-dimensional game, where Eve has three moves and Adam has two moves, is illustrated in Figure 1.1.



Adam's moves: $\{(1,2),(2,0)\}$

Eve's moves: $\{(2,2),(1,4),(3,0)\}$

Figure 1.1: An example of a $\mathbb{Z}^2$-VAS game. Eve (circle) has a winning strategy from the vectors in the lattice spanned by vectors $(-3,-4)$ and $(-4,-2)$.

Our main contribution is filling the decidability gaps for $\mathbb{Z}^d$-VASS and $\mathbb{Z}^d$-VAS games. We show that checking which player has a winning strategy in $\mathbb{Z}^2$-VAS games is undecidable and EXPSPACE-complete in one-dimensional $\mathbb{Z}$-VASS games. Previously, it has been proved that deciding the winner in one-dimensional $\mathbb{Z}$-VAS games, where integers are given in binary, is EXPTIME-complete [9] and undecidable starting from dimension three [137], and that two-dimensional $\mathbb{Z}$-VASS games are undecidable [137].

We also present three variants of low-dimensional Attacker-Defender games (i.e., word games, matrix games and braid games) for which it is undecidable to determine whether one of the players has a winning strategy. In addition, the proof incorporates a new language theoretical result about weighted automata on infinite words that can be efficiently used in the context of other reachability games.

While we are mostly focusing on two-player games, we also consider problems in single-player games. As there is no opposing player, the question of existence of a winning strategy is simply whether a particular configuration of the system can be reached from a given initial configuration of the system.

Apart from the abovementioned decidability and complexity results in different Attacker-Defender games, this thesis presents a collection of novel techniques and reductions that could be applied further to prove similar results in other games.

1.1 Thesis outline

The thesis consists of three independent parts. Although each chapter can be read as a separate entity, they are all devoted to the analysis of computational games with reachability objectives. The main model of the thesis are Attacker-Defender games, which are turn-based zero-sum games between two players, Adam and Eve, with reachability objective for Eve. The main question that we study is whether Eve has a winning strategy for the given Attacker-Defender game. Whenever the problem is decidable, we aim to find tight complexity bounds. We consider various Attacker-Defender games, both with internal states and without, and with vastly different moves. In Chapter 3, we consider stateless games where moves are words, matrices and braids, while in the following chapters, we focus on Attacker-Defender games, where moves are integer vectors, that is, games on $\mathbb{Z}$-VAS. We also consider $\mathbb{Z}$-VAS games where the players have an internal state structure.

Even though the games are different, and so are subsequent results, the main method used in proofs is similar. The broad idea is to simulate another computational model or a game and ensure, via the way the game is constructed, that unfaithful simulation has a predetermined outcome. Obviously, different models are simulated to achieve different results. To prove undecidability, we have simulated such models as the infinite Post correspondence problem and two-counter Minsky machines. In order to prove complexity bounds, we simulated models with desired decidable properties, such as alternating pushdown systems, one-dimensional counter reachability games and linear-bounded automata.

In Chapter 2, we introduce the definitions and notations, as well as few auxiliary results, used in subsequent chapters.

The first part, Chapter 3, is based on a conference paper [82] and its journal version [83]. The main result of the chapter is that the universality problem is undecidable for weighted automata on infinite words. The construction is rather involved and provides a new non-standard encoding of the infinite Post correspondence problem into the universality problem. Then we apply the main result to several games on mathematical objects that we call Attacker-Defender games, and show that it is undecidable to check whether one of the players has a winning strategy. The games we are considering are word games, where players' concatenate words over a group alphabet, matrix games, where players transform a given vector by multiplying it with matrices, and braid games, where players braid and unbraid a given braid. After each game, we provide an example to illustrate the main principles of the game.

The second part consists of three chapters with closely related games. We move on from more complex computational environments of Chapter 3 and consider games on integer vector addition systems. That is, Attacker-Defender games played on the integer lattice $\mathbb{Z}^d$.

First, in Chapter 4, which is based on a conference paper [123], we consider one-dimensional $\mathbb{Z}$-VASS games and show that deciding who wins the game is an EXPSPACE-complete problem. Noticing an interesting complexity gap between games with states and without states, we study flat $\mathbb{Z}$-VASS games and prove that, in dimension one, deciding the winner is EXPTIME-complete.

The following chapter is based on a conference paper [125] and is on two-dimensional $\mathbb{Z}$-VAS games. First, we construct a two-dimensional $\mathbb{Z}^2$-VASS game that follows the computation of a two-counter Minsky machine and show that it is undecidable which player has a winning strategy in $\mathbb{Z}^2$-VASS games. After that, we map the states and state transitions into integers and embed them into the least significant digits in vectors of a two-dimensional $\mathbb{Z}$-VAS game, showing that also the stateless two-dimensional $\mathbb{Z}$-VAS games are undecidable.

While studying $\mathbb{Z}^d$-VASS games, we noticed interesting diversity of the complexity arising from the differences in the state structure of the players. Motivated by this, in Chapter 6, we consider a different limitation to the game. We restrict the number of times one of the players can play a move from his or her move set. We show that deciding the winner in a $\mathbb{Z}^d$-VASS game where Adam can play limited number of times is NP-complete, while the dual case, i.e., when Eve's moves are limited, is in PTIME. The chapter is based on yet unpublished work.

The final part of the thesis consists of Chapter 7, in which we consider single-player reachability games. First, we prove that the reachability problem for polynomial iteration is PSPACE-complete. We also consider the multidimensional case and prove undecidability for three-dimensional polynomials. The first half of the chapter is based on a conference paper [124]. The second half of the chapter is based on unpublished work and is on the identity problem for matrices. We show that there is no embedding from pairs of words into 3×3 integral matrices with determinant one. This strongly suggests that the identity problem is decidable, as most of known undecidability proofs in matrix semigroups rely on an encoding of the PCP. Motivated by this, we consider the identity problem for the Heisenberg group, that is, the group of upper-triangular matrices with ones on the main diagonal, and prove that the problem is decidable in polynomial time for 3×3 matrices and in exponential time for larger dimensions.

We conclude the thesis with some final remarks and open problems.

1.2 Overview of related research

In this section, we highlight known results for several variants of computational games with different winning objectives.

Two-player games can be classified according to the winning conditions. The winning conditions can be either qualitative or quantitative, or a combination of both. Qualitative objectives require that a winning play of a player satisfies some (Boolean) property. Perhaps the most used qualitative objective is the parity objective, where nodes of the arena are assigned colours (integers) and in a winning play of Eve, the smallest colour that appears infinitely often is even. Parity games have strong connections, among others, with μ-calculus, modal logics and tree automata [17, 64, 142, 147].

First, it was proven that parity games can be solved in exponential time [119, 148]. Later, the complexity bound was first improved to $\mathsf{NP} \cap \mathsf{coNP}$ [65] and then to $\mathsf{UP} \cap \mathsf{coUP}$ [90]. Since then, the exact complexity has been improved [91] and at the moment, the state-of-art complexity is due to Calude et al. [33]. Despite the best efforts, the open question of [64] remains — can parity games be solved in polynomial time?

As parity games are highly applicable, there has been a substantial study in different variants of the games, mostly resulting in PTIME complexity to decide which player wins. Different underlying graphs were considered in [60, 71] and authors showed that for these classes of graphs, the game is solvable in PTIME. In [143], the games were considered with significantly more vertices in the graph than priorities, which naturally arise from different translations of games into parity games.

For the quantitative objectives, often the energy objective is considered. In games with energy objectives, the energy level of a system is represented by an integer vector. Eve aims to keep the energy levels positive, while Adam tries to reduce some component below zero. Energy games can be applied in various settings, such as, in weak simulation between a finite state system and a Petri net [3, 57] or in model-checking for resource-bounded logic $\mathrm{RB} \pm \mathrm{ATL}^*$ in artificial intelligence [5, 6].

There are several problems for energy games that have been considered. In [29], it was shown that deciding whether an initial credit exists that would allow Eve to maintain non-negative energy levels is a coNP-complete problem. On the other hand, deciding the winner for a given initial credit is a 2-$\mathsf{EXPTIME}$-complete problem [57, 92]. If the dimension

is fixed, then both problems can be solved in pseudo-polynomial time [92].

In energy parity games [38] winning objectives are a combination of the aforementioned qualitative and quantitative objectives. In [41,42], it was proved that deciding which player has a winning strategy for games with arbitrary initial credit remains a coNP-complete problem. With given initial credit, the problem was shown to be decidable in [3]. Later, 2-EXPTIME-hard lower bound was shown in [57]. The matching upper bound was recently proved in [50]. Moreover, if the dimension and number of priorities are fixed, the winner can be determined in pseudo-polynomial time.

Energy objectives are not sufficient to model some rather natural systems. While it is easy to model consumption and replenishment of resources in a system, it is often important to know the average behaviour of the resource in a long run. To this end different payoff functions were studied. Intuitively, to each sequence of edges traversed in the game, a payoff function is applied to compute the value of the play. Typical payoff functions are mean-payoff, (total) sum, and discounted sum. In the associated decision problem, we are given a threshold ν and are asked whether Eve has a strategy ensuring that all plays have payoff of at least ν. Let $w(e_i)$ denote the integer label of an edge taken during ith turn of the game, then in one-dimensional mean-payoff games, we are interested whether Eve can ensure that $\liminf_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} w(e_i) \geq \nu$ holds in all plays according to her strategy. In total sum, the payoff function is slightly different and Eve tries to ensure that $\liminf_{n\to\infty} \sum_{i=1}^{n} w(e_i) \geq \nu$. That is, the payoff is the sum of the edges taken during a play rather than the mean of the sum.

For one-dimensional mean-payoff games, the problem was proved to be decidable in NP ∩ coNP [149] and then improved to UP ∩ coUP by Jurdzinski [90], and for total sum games in EXPTIME [30,31]. The games generalise to higher dimensions in a natural way — the payoff function is simply applied to each component. In mean-payoff games, the problem is coNP-complete if the dimension is not fixed [39], while for total payoff games, the problem is undecidable starting from dimension five [40]. On the other hand, if the number of dimensions and the maximal absolute value of weights are fixed, then multidimensional mean-payoff games can be solved in PTIME [43,44].

Games with payoffs provide a fruitful domain for research. In [145], multidimensional mean-payoffs, where in each dimension, rather than having only $\geq$ as the comparison operation, other comparison operations, such as $\leq$, $<$ and $>$, are allowed, were studied. This generalisation leads to undecidability of determining whether a winning strategy exists in 10-dimensional games. In [89], the games with various payoff functions were considered,

but rather than considering whether the value of a play is over the threshold, the authors are considering whether the value lies within the interval union. In [28], a different kind of behaviour of mean payoff games was studied. The authors, introduced a notion of stability, to distinguish runs by fluctuations from the mean. The main result is that the problem is decidable and PSPACE-hard.

Our main subject of study, $\mathbb{Z}^d$-VAS games, are subfamily of counter reachability games, where the game is played on a graph with vertices partitioned between players. It has been proved that deciding the winner in two-dimensional counter reachability games is undecidable [138]. Our result can be seen as strengthening of this as our arena is a graph without self-loops and with one vertex for each player, i.e., both players are stateless.

In [1, 2, 29], VASS games, where the game is played on a graph and counters are always positive, were considered. It was proven that already in two dimensions it is undecidable who wins if Eve's goal is to reach a particular vertex with counter $(0, 0)$. On the other hand, if it can be any vertex, then the problem is in $(k-1)$-EXPTIME for a game with k counters. Later, the result was improved to PTIME for $k = 2$ [37]. The counter reachability games of [138] are VASS games, where the possible counter values were extended to all integers. Hunter considered the variants of games, where updates on the counters are done in binary, and showed that one-dimensional games are EXPSPACE-complete [88].

The proofs of undecidability of VASS games and counter reachability in two dimensions in [1, 29] use the state structure of the game to embed the state structure of a two-counter machine. In this sense, our result on $\mathbb{Z}^2$-VASS games is comparable, as Eve simulates the state transitions of a two-counter machine with her underlying automaton. On the other hand, the stateless game is essentially different as we have to represent state transitions with integers. When simulating a two-counter machine, it is possible for Eve to make a wrong move and then Adam is able to ensure his victory from this point onward. In $\mathbb{Z}^d$-VAS games, Eve's state is dependent only on her previous moves, while in VASS games or counter reachability games, Adam's moves effect which state Eve enters. Because of this, Adam's cheat catching ability is implemented in a different way.

In Chapter 3, the considered model of automaton is closely related to *integer weighted finite automata* as defined in [79] and [7], where finite automata are accepting finite words and have additive integer weights on the transitions. In [79], it was shown that the universality problem is undecidable for integer weighted finite automata on finite words by reduction from the Post correspondence problem. In the game scenario it is important to define acceptance of infinite words (which represent infinite plays in games) by considering

finite prefixes reaching a target value. On the other hand, non-acceptance means that there exists an infinite computational path where none of the finite prefixes reach the target value. Then the universality for weighted automata over infinite words is the property ensuring that all infinite words are accepted (i.e., eventually reach a target in a computation path).

The models similar to our polynomial iteration of Chapter 7 have been studied before. In [24], polynomial iteration in $\mathbb{Q}$ was studied and the reachability problem was proved to be decidable using p-adic norms. Polynomials over $\mathbb{Q}$ are significantly harder to analyse than polynomials over $\mathbb{Z}$, as in a finite interval $[a, b]$, there might be an infinite number of reachable values.

Multidimensional linear polynomial iteration has been considered from a different aspect. The vector reachability problem for d-dimensional matrices over $\mathbb{F}$, where $\mathbb{F} = \mathbb{Z}, \mathbb{Q}, \mathbb{C}, \ldots$, studies whether for given two vectors $\mathbf{x}_0, \mathbf{x}_f$ and a set of matrices $\{M_1, \ldots, M_k\} \subseteq \mathbb{F}^{d\times d}$, there exists a finite sequence of matrices such that $M_{i_1} \cdots M_{i_j} \mathbf{x}_0 = \mathbf{x}_f$. Since transforming a vector by a matrix can be expressed as a system of linear equations, the multidimensional linear polynomial iteration can be seen as a vector reachability problem. The main difference from our consideration is that we consider only polynomials of the form

$$p(x_1, \ldots, x_d) = (p_1(x_1), \ldots, p_d(x_d))$$

for some univariate polynomials $p_i(x)$, while the polynomials in the vector reachability problem are of the form

$$p(x_1, \ldots, x_d) = (a_{11}x_1 + \ldots + a_{1d}x_d, \ldots, a_{d1}x_1 + \ldots + a_{dd}x_d).$$

The vector reachability problem has been proven to be undecidable for six three-dimensional integer matrices in [81] and for two 11-dimensional integer matrices in [84].

In [12], the authors studied reachability of a point in $\mathbb{Q}^2$ by two-dimensional affine transformations. They proved that the problem is undecidable already for five such affine polynomials. The affine transformations used are of the form

$$p(x, y) = (q_1x + q_2y + q_3, q_4x + q_5y + q_6).$$

The above mentioned undecidability results relied on the undecidability of the Post correspondence problem with seven pairs of words and having particular structure known as Claus instances [49]. The state-of-art bound on the number of pairs of words is five [122],

which could result in lower bounds on the number of matrices and linear transformations.

The polynomial iteration can be also considered as piecewise maps. That is, a polynomial $p(x)$ is applicable only when $x \in [a, b)$ for some $a, b \in \mathbb{Z} \cup \{\pm\infty\}$. Piecewise maps and related reachability problems have been studied extensively [16, 97, 104]. The problem is undecidable for two-dimensional piecewise affine maps. The decidability of the reachability problem for one-dimensional piecewise affine maps is an open problem even when there are only two intervals [24, 104]. On the other hand, for more general updates the problem is undecidable. For example, if the updates are based on the elementary functions $\{x^2, x^3, \sqrt{x}, \sqrt[3]{x}, 2x, x+1, x-1\}$ or on rational functions of the form $p(x) = \frac{ax^2+bx+c}{dx+e}$, where the coefficients are rational numbers [105], then the problem is undecidable.

There are many systems and models, which are represented by matrices, while the behaviour of the systems is represented by matrix products. Their analysis and prediction are the challenging problems that appear in verification, control theory questions, biological systems, etc. [20, 21, 46, 48, 70, 98, 121, 126–128]. Many nontrivial algorithms for solving decision problems on matrix semigroups are developed, when considering matrices under different constraints like the dimension of matrices, number of matrices in the generator set, or considering specific subclasses of matrices: e.g., the general class of commutative matrices [10], non-commutative case of row-monomial matrices [112] or various subclasses of 2×2 matrix semigroups generated by non-singular integer matrices [134], upper-triangular integer matrices [87], matrices from the special linear group [11, 45], etc.

Despite visible interest in this research domain, we still see a significant lack of algorithms and complexity results for answering decision problems in matrix semigroups. Many computational problems for matrix (semi)groups are computationally hard starting from dimension two and very often become undecidable from dimensions three or four even in the case of integer matrices. The central decision problem in matrix semigroups is the membership problem, which was originally considered by A. Markov in 1947 [115]. Let $S = \langle G \rangle$ be a matrix semigroup finitely generated by a generating set of square matrices G. The *membership problem* is to decide whether or not a given matrix M belongs to the matrix semigroup S. By restricting M to be the identity matrix we call the problem the *identity problem*. The identity problem is computationally equivalent to another fundamental problem – the *subgroup problem* (i.e. to decide whether a semigroup contains a subgroup) as any subset of matrices, which can form a product leading to the identity also generate a group [45].

The decidability status of the identity problem was unknown for a long time for matrix

semigroups of any dimension, see Problem 10.3 in "Unsolved Problems in Mathematical Systems and Control Theory" [21], but it was shown in [14] to be undecidable for 48 matrices from $\mathbb{Z}^{4\times4}$ by proving that the identity correspondence problem (a variant of Post correspondence problem over a group alphabet) is undecidable, and embedding pairs of words over free group alphabet into $\mathrm{SL}(4,\mathbb{Z})$ as two blocks on the main diagonal and by a morphism f as follows

$$f(a)=\begin{pmatrix}1&2\\0&1\end{pmatrix},\quad f(a^{-1})=\begin{pmatrix}1&-2\\0&1\end{pmatrix},\quad f(b)=\begin{pmatrix}1&0\\2&1\end{pmatrix},\quad f(b^{-1})=\begin{pmatrix}1&0\\-2&1\end{pmatrix}.$$

In the seminal paper of Paterson in 1970, see [130], an injective morphism from pairs of words in alphabet $\Sigma=\{a,b\}$ into 3×3 integral matrices,

$$g(u,v)=\begin{pmatrix}n^{|u|}&0&0\\0&n^{|v|}&0\\\sigma(u)&\sigma(v)&1\end{pmatrix}$$

(where σ represents each word as an n-adic number) was used to prove undecidability of mortality and which later led to many undecidability results of matrix problems in dimension three, e.g. [35, 36, 81]. Finding new injective morphisms is hard, but having them gives an opportunity to prove new undecidability results.

In 1999, Cassaigne, Harju and Karhumäki significantly boosted the research on finding algorithmic solutions for 2×2 matrix semigroups by showing that there is no injective semigroup morphism from pairs of words over any finite alphabet (with at least two elements) into complex 2×2 matrices [36]. This result led to substantial interests in finding algorithmic solutions for such problems as the identity problem, mortality, membership, vector reachability, freeness, etc. for 2×2 matrices.

For example, in 2007 Gurevich and Schupp [76] showed that the membership problem is decidable in polynomial time for the finitely generated subgroups of the modular group and later in 2017 Bell, Hirvensalo and Potapov proved that the identity problem for a semigroup generated by matrices from $\mathrm{SL}(2,\mathbb{Z})$ is NP-complete by developing a new effective technique to operate with compressed word representations of matrices and closing the gap on complexity improving the original EXPSPACE solution proposed in 2005 [45]. The first algorithm for the membership problem, which covers the cases beyond $\mathrm{SL}(2,\mathbb{Z})$ and $\mathrm{GL}(2,\mathbb{Z})$, has been proposed in [134] and provides the solution for a semigroup generated

by non-singular 2×2 integer matrices. Later, these techniques have been applied to build another algorithm to solve the membership problem in $\mathrm{GL}(2, \mathbb{Z})$ extended by singular matrices [135]. The current limit of decidability is standing for 2×2 matrices, which are defined over hypercomplex numbers (quaternions), for which most of the problems have been shown to be undecidable in [13] and correspond to reachability problems for 3-sphere rotation.

1.3 Author's publications

The peer-reviewed publications and submitted manuscripts of the author are listed in Table 1.1.

The results of this thesis have been presented at Automatic Sequences (2015), British Colloquium of Theoretical Computer Science (BCTCS 2016), Computability in Europe (CiE 2015), Finnish Mathematical Days (2016), Highlights of Logic, Games and Automata (2016), Language and Automata Theory and Applications (LATA 2015), Mathematical Foundations of Computer Science (MFCS 2016), Reachability Problems (RP 2016, RP 2017), Russian Finnish Symposium on Discrete Mathematics (RuFiDiM 2014), SET for BRITAIN (2016), and several seminars in the Department of Computer Science at the University of Liverpool.

Title	Authors	Venue
On decidability and complexity of low-dimensional robot games	R. Niskanen, I. Potapov, J. Reichert	submitted manuscript
On robot games of degree two [86]	V. Halava, R. Niskanen, I. Potapov	LATA 2015 [3]
On the identity problem for the special linear group and the Heisenberg group [95]	S-K. Ko, R. Niskanen, I. Potapov	submitted manuscript
Reachability problem for polynomial iteration is PSPACE-complete [124]	R. Niskanen	RP 2017 [4]
Robot games with states in dimension one [123]	R. Niskanen	RP 2016 [5]
Undecidability of two-dimensional robot games [125]	R. Niskanen, I. Potapov, J. Reichert	MFCS 2016 [6]
Weighted automata on infinite words in the context of Attacker-Defender games [82]	V. Halava, T. Harju, R. Niskanen, I. Potapov	CiE 2015 [7]
Weighted automata on infinite words in the context of Attacker-Defender games [83]	V. Halava, T. Harju, R. Niskanen, I. Potapov	Information and Computation 2017

Table 1.1: Author's publications on which the thesis is based.

[3] Language and Automata Theory and Applications – 9th International Conference
[4] Reachability Problems – 11th International Workshop
[5] Reachability Problems – 10th International Workshop
[6] 41st International Symposium on Mathematical Foundations of Computer Science
[7] Evolving Computability – 11th Conference on Computability in Europe

Chapter 2

Preliminaries

In this chapter, we introduce the notation and definitions used throughout the thesis. Mathematical abstractions of games are often described in the language of combinatorics on words, automata theory and other computational models.

First, we introduce basic definitions on words and matrices. Then, we consider different computational models used in our proofs, most notably the integer weighted automata on infinite words that we use in proofs of Chapter 3. In the third section, we look at Attacker-Defender games, which is the main game model used in the thesis. In particular, we introduce different variants of Attacker-Defender games, for which we prove either undecidability of checking for existence of a winning strategy or show the complexity of the problem. In the final section, Section 2.5, we consider properties of a particular instance of the infinite Post correspondence problem used in Chapter 3.

2.1 Words, matrices and the PCP

Basic definitions We denote the sets of integers, non-positive integers and non-negative integers (that is, natural numbers) by $\mathbb{Z}$, $\mathbb{Z}^-$ and $\mathbb{Z}^+$ respectively. The sets of rational, real and complex numbers are denoted by $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$. Unless stated otherwise, all numbers are encoded in binary. By $\mathbf{0}_d$ we denote the d-dimensional zero vector. An open interval (a, b) is a subset of $\mathbb{Z}$ containing all the integers larger than a and smaller than b. A closed interval $[a, b]$ is $(a, b) \cup \{a, b\}$ and half-open intervals are defined similarly. Let $X \subseteq \mathbb{Z}$. By $X + d$ and dX, where $d \in \mathbb{Z}$, we denote the sets $\{x + d \mid x \in X\}$ and $\{dx \mid x \in X\}$. The disjoint union of two disjoint sets X and Y (i.e., $X \cap Y = \emptyset$) is denoted by $X \sqcup Y$.

By $\mathbb{Z}[x]$ we denote the ring of polynomials with integer variable x. A polynomial $p(x) \in \mathbb{Z}[x]$ is $p(x) = a_n x^n + \ldots + a_1 x + a_0$, where $a_i \in \mathbb{Z}$ and $n \geq 0$. We represent polynomials in sparse encoding by a sequence of pairs $(i, a_i)_{i \in I}$, where $I = \{i \in \{0, \ldots, n\} \mid a_i \neq 0\}$. Deciding whether for a given $y \in \mathbb{Z}$, the polynomial $p(y)$ evaluates to a positive number can be done in polynomial time [58].

In our encoding of Section 7.1, we use the Chinese remainder theorem to find the unique solution to a system of linear congruences. That is, for given pairwise co-prime positive integers $n_1, \ldots, n_k$ and $b_1, \ldots, b_k \in \mathbb{Z}$, the system of linear congruences $x \equiv b_i \mod n_i$ for $i = 1, \ldots, k$ has a unique solution modulo $n_1 \cdots n_k$. Recall that a residue class b modulo n is the set of integers $n\mathbb{Z} + b = \{\ldots, b - n, b, b + n, \ldots\}$.

Words A *semigroup* is a set equipped with an associative binary operation. Let S be a semigroup and Σ be a subset of S. We say that S is *generated* by Σ of S if each element of S can be expressed as a composition of elements of Σ. In this case, we call Σ the *generating set* of S. Given an *alphabet* $\Sigma = \{a_1, a_2, \ldots, a_m\}$, a finite *word* is an element of semigroup Σ^*. The *empty word* is denoted by ε. The length of a finite word u is denoted by $|u|$ and $|\varepsilon| = 0$. A word $u \in \Sigma^*$ is a *prefix* of $v \in \Sigma^*$, denoted by $u \leq v$, if $v = uw$ for some $w \in \Sigma^*$. If u and w are both nonempty, then the prefix u is called *proper*, denoted by $u < v$.

An *infinite word* w over a finite alphabet Σ is an infinite sequence of letters, $w = a_{i_1} a_{i_2} a_{i_3} \cdots$, where $a_{i_j} \in \Sigma$ is a letter for each $j = 1, 2, \ldots$. We denote the set of all infinite words over Σ by Σ^ω. A *prefix* of an infinite word $w \in \Sigma^\omega$ is a finite word $p \in \Sigma^*$ such that $w = pw'$, where $w' \in \Sigma^\omega$. This is also denoted by $p \leq w$.

By $w[i]$ we denote the ith letter of a word w, i.e., $w = w[1]w[2] \cdots$.

Later, in Chapter 3, finite words will be denoted by u, v, infinite words by w and single letters by a, b, c, x, y, z.

Let $\Gamma = \{a_1, a_2, \ldots, a_m, a_1^{-1}, a_2^{-1}, \ldots, a_m^{-1}\}$ be a generating set of a free group $\mathrm{FG}(\Gamma)$. The elements of $\mathrm{FG}(\Gamma)$ are all *reduced* words over Γ, i.e., words not containing $a_i a_i^{-1}$ or $a_i^{-1} a_i$ as a subword. In this context, we call Γ a finite *group alphabet*, i.e., an alphabet with an involution. The multiplication of two elements (reduced words) $u, v \in \mathrm{FG}(\Gamma)$ corresponds to the unique reduced word of the concatenation uv. This multiplication is called *concatenation* throughout the thesis. Later in the encoding of words over a group alphabet we denote a^{-1} by $\overline{a}$ and the alphabet of inverse letters is denoted as $\Sigma^{-1} = \{a^{-1} \mid a \in \Sigma\}$.

In the next lemma, we present an encoding from an arbitrary group alphabet to a binary group alphabet used in Section 3.2 and Section 7.4. The result is crucial as it allows us to

present the results of these sections over the smallest domain.

Lemma 2.1 (Birget, Margolis [19]). *Let* $\Gamma = \{z_1, \ldots, z_\ell, \overline{z_1}, \ldots, \overline{z_\ell}\}$ *be a group alphabet and* $\Gamma_2 = \{c, d, \bar{c}, \bar{d}\}$ *be a binary group alphabet. Define the mapping* $\alpha : \Gamma \to \mathrm{FG}(\Gamma_2)$ *by:*

$$\alpha(z_i) = c^i d \bar{c}^i, \qquad \alpha(\overline{z_i}) = c^i \bar{d} \bar{c}^i,$$

where $1 \leq i \leq \ell$. *Then* α *is a monomorphism, that is, injective morphism. Note that* α *can be extended to domain* $\mathrm{FG}(\Gamma)$ *in the usual way.*

Matrices In this thesis, we consider games with different environments and moves, including games where the configuration is a point in a d-dimensional space and moves are linear transformations that we express as matrices. In our considerations, the most prominent matrix groups are the *special linear group* and the *Heisenberg group*.

The special linear group is $\mathrm{SL}(d, \mathbb{K}) = \{M \in \mathbb{K}^{d \times d} \mid \det(M) = 1\}$, where $\mathbb{K} = \mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}, \ldots$. The *identity matrix* is denoted by $\boldsymbol{I}_d$ and the *zero matrix* is denoted by $\boldsymbol{O}_d$. The *Heisenberg group* $\mathrm{H}(3, \mathbb{K})$ is formed by the 3×3 matrices of the form

$$M = \begin{pmatrix} 1 & a & c \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix},$$

where $a, b, c \in \mathbb{K}$. It is easy to see that the Heisenberg group is a non-commutative subgroup of $\mathrm{SL}(3, \mathbb{K})$. We can consider the Heisenberg group as a set of all triples with the following group law:

$$(a_1, b_1, c_1) \otimes (a_2, b_2, c_2) = (a_1 + a_2, b_1 + b_2, c_1 + c_2 + a_1 b_2).$$

By $\psi(M)$ we denote the triple $(a, b, c) \in \mathbb{K}^3$ which corresponds to the upper-triangular coordinates of M. Let M be a matrix in $\mathrm{H}(3, \mathbb{K})$ such that $\psi(M) = (a, b, c)$. We define the *superdiagonal vector* of M to be $\vec{v}(M) = (a, b)$. Given two vectors $\mathbf{u} = (u_1, u_2)$ and $\mathbf{v} = (v_1, v_2)$, the *cross product* of $\mathbf{u}$ and $\mathbf{v}$ is defined as $\mathbf{u} \times \mathbf{v} = u_1 v_2 - u_2 v_1$. Any two vectors are said to be *parallel* if the cross product is zero.

The Heisenberg group can also be defined in higher dimensions. The Heisenberg group of dimension d over $\mathbb{K}$ is denoted by $\mathrm{H}(d, \mathbb{K})$ and is the group of square matrices in $\mathbb{K}^{d \times d}$ of

the following form:

$$\begin{pmatrix} 1 & \mathbf{a}^T & c \\ 0 & \boldsymbol{I}_{d-2} & \mathbf{b} \\ 0 & 0 & 1 \end{pmatrix},$$

where $\mathbf{a}, \mathbf{b} \in \mathbb{K}^{d-2}, c \in \mathbb{K}$.

As we have considered for the Heisenberg group in dimension three, we can also consider the Heisenberg group in dimension d for any integer $d \geq 3$ as a set of all triples with the following group law: $(\mathbf{a}_1, \mathbf{b}_1, c_1) \otimes (\mathbf{a}_2, \mathbf{b}_2, c_2) = (\mathbf{a}_1 + \mathbf{a}_2, \mathbf{b}_1 + \mathbf{b}_2, c_1 + c_2 + \mathbf{a}_1 \cdot \mathbf{b}_2)$, where $\mathbf{a}_1, \mathbf{a}_2, \mathbf{b}_1, \mathbf{b}_2 \in \mathbb{K}^{d-2}$ and $\mathbf{a}_1 \cdot \mathbf{b}_2$ is the dot product of vectors $\mathbf{a}_1$ and $\mathbf{b}_2$.

We extend the function ψ to d-dimensional Heisenberg group: For a matrix M, $\psi(M)$ is the triple $(\mathbf{a}, \mathbf{b}, c) \in (\mathbb{K}^{d-2})^2 \times \mathbb{K}$ which corresponds to the upper-triangular coordinates of M.

Next, we prove a simple necessary and sufficient condition for commutation of two matrices from the Heisenberg group.

Lemma 2.2. *Let M_1 and M_2 be two matrices from the Heisenberg group $\mathrm{H}(d, \mathbb{K})$ and $\psi(M_i) = (\mathbf{a}_i, \mathbf{b}_i, c_i)$ for $i = 1, 2$. Then $M_1 M_2 = M_2 M_1$ holds if and only if $\mathbf{a}_1 \cdot \mathbf{b}_2 = \mathbf{a}_2 \cdot \mathbf{b}_1$.*

Proof. The product $M_1 M_2$ has $c_1 + c_2 + \mathbf{a}_1 \cdot \mathbf{b}_2$ in the upper-right corner whereas $M_2 M_1$ has $c_1 + c_2 + \mathbf{a}_2 \cdot \mathbf{b}_1$. The other coordinates are identical as we essentially add numbers in the same coordinate. It is easy to see that the two products are equivalent if and only if $\mathbf{a}_1 \cdot \mathbf{b}_2 = \mathbf{a}_2 \cdot \mathbf{b}_1$ holds. □

Note that, in the Heisenberg group of dimension three, the condition of Lemma 2.2 can be stated as superdiagonal vectors of M_1 and M_2 being parallel.

Post correspondence problem and its variants The *Post correspondence problem* (PCP) is a famous undecidable problem introduced by Emil Post in 1946 [131]. The PCP is used in a variety of settings due to its simple formulation. For us, the infinite variant of the PCP is crucial in proofs of Chapter 3, where we prove that for several Attacker-Defender games, it is undecidable whether the winning strategy exists for one of the players. We require the infinite variant as the games we consider are of infinite duration. Additionally, in Section 7.4, we use the PCP to reduce the number of generators needed to prove the undecidability of the identity problem for $\mathrm{SL}(4, \mathbb{Z})$.

There are several equivalent ways to formulate the PCP. We define it using morphisms. An *instance* of the PCP consists of two morphisms $g, h : \Sigma^* \to B^*$, where Σ and B are alphabets. A nonempty word $u \in \Sigma^*$ is a *solution* of an instance (g, h) if it satisfies $g(u) = h(u)$. The problem is undecidable for all domain alphabets Σ with $|\Sigma| \geq 5$ [122]. The cardinality of the domain alphabet Σ is said to be the *size* of the instance and is denoted by n_p.

The *infinite Post correspondence problem* (ωPCP) is a natural extension of the PCP. An infinite word w is a *solution* of an instance (g, h) of the ωPCP if for every finite prefix p of w either

$$h(p) < g(p) \text{ or } g(p) < h(p). \tag{2.1}$$

In the ωPCP, it is asked whether or not a given instance has an infinite solution. Note that in our formulation, prefixes have to be proper. It was proven in [80] that the problem is undecidable for all domain alphabets Σ with $|\Sigma| \geq 9$, and it was improved to $|\Sigma| \geq 8$ in [61], for a variant of the problem, where the prefixes in (2.1) do not have to be proper. However, it is easy to see that adding a new letter α to the alphabets and desynchronising the morphisms g, h, gives us a solution where all prefixes have to be proper. That is, we add α to the left of each letter in the image under h, to the right of each letter in the image under g and $g(\alpha) = a\alpha, h(\alpha) = a$ for some $a \in \Sigma$. Now the solution has to start with the letter α, α has to appear exactly once, and the images cannot be of equal lengths because the image under g ends with α but not under h. Note that, in fact, both constructions already have this desynchronising property. See Section 2.5 and [61, 80] for more details on the morphisms g and h.

Consider an infinite word $w \in \Sigma^\omega$ that is not a solution of the ωPCP. It is clear that there exists (at least one) integer i such that $g(w)[i] \neq h(w)[i]$. That is, the letter in the position i of $g(w)$ is different from the letter in the position i of $h(w)$. We call this kind of mismatch *an error*.

The *identity correspondence problem* (ICP) [14] is another variant of the PCP which asks whether a finite set of pairs of words over a group alphabet can generate the identity pair by a sequence of concatenations. That is, for two morphisms $g, h : \Sigma^* \to \mathrm{FG}(\Gamma)$, the ICP asks if there exists a word $u \in \Sigma^*$ such that $g(u) = h(u) = \varepsilon$. Bell and Potapov [14] proved that the ICP is undecidable by a constructive reduction from the restricted PCP[8] and showed

[8] In the restricted PCP, the solution starts with a letter a_1, ends with a_n and these letters are used

that the undecidability bound for the ICP is $8(n_r - 1)$, where n_r is the undecidability bound of the restricted PCP. The current bound for the ICP is 48.

Example 2.3. Let $\Sigma = \{1, 2, 3\}$ and $B = \{a, b\}$. Define morphisms $g, h : \Sigma^* \to B^*$ as

$$\begin{aligned} g(1) &= ab, & h(1) &= abb, \\ g(2) &= bb, & h(2) &= baa, \\ g(3) &= aaa, & h(3) &= aa. \end{aligned}$$

If (g, h) is considered as an instance of the PCP, then the word $u = 1233$ is a solution. Indeed,

$$g(1233) = \overbrace{\underbrace{a}_{h(1)}\ \underbrace{b}_{h(1)}}^{g(1)}\ \overbrace{\underbrace{b}_{h(2)}}^{g(2)}\ \overbrace{\underbrace{a}_{h(3)}\ \underbrace{a}_{h(3)}}^{g(3)}\ \overbrace{\underbrace{a}_{h(3)}\ \underbrace{a}_{h(3)}}^{g(3)} = h(1233).$$

If (g, h) is considered as an instance of the ωPCP, then an infinite repetition of u, u^ω, is not a solution because $h(u)$ and $g(u)$ are not proper prefixes of each other. However, the instance has a solution, $w = 3^\omega$, as for any finite prefix p of w, $h(p) < g(p)$.

2.2 Models of computation

When considering some Attacker-Defender games, we use different computation models to prove both undecidability of checking for existence of a winning strategy and to prove upper and lower bounds if the problem is decidable. Next, we define integer weighted automata on infinite words, and the universality problem, used to prove undecidability in word games, matrix games on vectors and braid games in Chapter 3. The universality problem is closely related to a winning strategy of Adam. Indeed, as Adam is the universal player, his goal is to show that *for all* plays his winning conditions are met, while in the terminology of automata, the universality is whether *all* words are accepted by the automaton.

Then we define two-counter machines used in the undecidability proofs of Chapter 5 and finally we conclude the section with polynomial register machines and linear-bounded automata, which are used in Section 7.1 to prove that single-player games with polynomial updates as moves are PSPACE-complete.

---

exactly once.

Weighted automata Let $\mathcal{A} = (Q, \Sigma, \sigma, q_0, F, \mathbb{Z})$ be a finite integer weighted automaton with the set of states Q, the finite alphabet Σ, the set of transitions $\sigma \subseteq Q \times \Sigma \times Q \times \mathbb{Z}$, the initial state q_0, the set of final states $F \subseteq Q$ and the additive group of integers $\mathbb{Z}$ with identity 0, that is $(\mathbb{Z}, +, 0)$, as weights. Note that while we restrict ourselves to the case where the weights of the automaton are elements of the additive group of integers $\mathbb{Z}$, we could define the model for any other group $(G, \cdot, \iota)$ as well. We write the transitions in the form $t = \langle q, a, p, z\rangle \in \sigma$. In a graphical presentation, a transition t is denoted by $q \xrightarrow{(a,z)} p$.

A *configuration* of $\mathcal{A}$ is any triple $[q, u, z] \in Q \times \Sigma^* \times \mathbb{Z}$. A configuration $[q, u, z_1]$ is said to *yield* a configuration $[p, ua, z_1 + z_2]$ if there is a transition $\langle q, a, p, z_2\rangle \in \sigma$. This is denoted by $[q, u, z_1] \models_{\mathcal{A}} [p, ua, z_1 + z_2]$. Let $\models^*_{\mathcal{A}}$ or simply $\models^*$, if $\mathcal{A}$ is clear from the context, be the reflexive and transitive closure of the relation $\models_{\mathcal{A}}$.

A finite word u is accepted by a weighted automaton if there is a computation path labelled by u such that weights of the transitions add up to zero. In [79], it was shown that the universality problem for finite words is undecidable. In order to analyse infinite runs in infinite-state games, we extend the model of weighted automata to infinite words.

Let $\pi = t_{i_0} t_{i_1} \cdots$ be an infinite path of transitions of $\mathcal{A}$, where $t_{i_j} = \langle q_{i_j}, a_{i_j}, q_{i_{j+1}}, z_j\rangle$ for $j \geq 0$ and $q_{i_0} = q_0$. We call such path π a *computation path*. Denote by $\mathcal{R}(\pi)$ the set of all reachable configurations following a path π. That is, for

$$\pi = \langle q_0, a_{i_0}, q_{i_1}, z_0\rangle \langle q_{i_1}, a_{i_1}, q_{i_2}, z_1\rangle \langle q_{i_2}, a_{i_2}, q_{i_3}, z_2\rangle \cdots ,$$

the set of reachable configurations is

$$\mathcal{R}(\pi) = \{[q_0, \varepsilon, 0], [q_{i_1}, a_{i_0}, z_0], [q_{i_2}, a_{i_0} a_{i_1}, z_0 + z_1], [q_{i_3}, a_{i_0} a_{i_1} a_{i_2}, z_0 + z_1 + z_2], \ldots\} .$$

Let us define a morphism $\|\cdot\| \colon \sigma^\omega \to \Sigma^\omega$ by setting $\|t\| = a$ if $t = \langle q, a, p, z\rangle$. Let π be a computation path for which $\|\pi\| = w$. We say that the computation path π *reads* the word w. Let $c = [q, u, z] \in \mathcal{R}(\pi)$ for some computation path π. The *weight* of the configuration c is $\gamma(c) = z$ and we say that c is in the state q. When a computation path π reading the word w is fixed, by the *weight of prefix* $\gamma(p)$, we denote the weight of the configuration $[q, p, z] \in \mathcal{R}(\pi)$, where $p < w$.

Let us now define the acceptance condition for weighted automata on infinite words. An infinite word $w \in \Sigma^\omega$ is accepted by $\mathcal{A}$ if there exists an infinite path π such that at least one configuration c in $\mathcal{R}(\pi)$ is in a final state and has weight $\gamma(c) = 0$. The language

accepted by $\mathcal{A}$ is

$$L(\mathcal{A}) = \{w \in \Sigma^\omega \mid \exists \pi \in \sigma^\omega \colon ||\pi|| = w \text{ and } \exists [q, u, 0] \in \mathcal{R}(\pi) \colon q \in F\}\,.$$

From the definition of the acceptance it follows that the automaton does not have deadlocks, i.e., states with no outgoing transitions. Note also that, for each $w \in \Sigma^\omega$, there exists a computation path in the automaton.

Universality problem The *universality problem* for automata over infinite words is to decide, given a weighted automaton $\mathcal{A}$, whether the language accepted by $\mathcal{A}$ is the set of all infinite words. In other words, whether or not $L(\mathcal{A}) = \Sigma^\omega$. The problem of *non-universality* is the complement of the universality problem, that is, whether or not $L(\mathcal{A}) \neq \Sigma^\omega$ or whether there exists a $w \in \Sigma^\omega$ such that, for every computation path π of w, all configurations $[q, u, z] \in \mathcal{R}(\pi)$ in a final state do not have zero weight.

Let us compare the universality problem for automata over finite and infinite words. Let $\mathcal{B}$ be a complete weighted automaton on finite words and by $\mathcal{A}$ we refer to the same automaton on infinite words. If $\mathcal{B}$ is universal on finite words, then it is easy to see that $\mathcal{A}$ is also universal on infinite words. Indeed, if $\mathcal{B}$ accepts Σ^* then in particular it accepts a and all infinite words starting with a are accepted by $\mathcal{A}$. But on the other hand, if $\mathcal{B}$ is not universal on finite words, then it does not follow that $\mathcal{A}$ is non-universal on infinite words as well. Consider the weighted automaton $(\{q_0, q_1\}, \{a\}, \sigma, q_0, \{q_0\}, \mathbb{Z})$, where $\sigma = \{\langle q_0, a, q_1, 1\rangle, \langle q_1, a, q_0, -1\rangle\}$, depicted in Figure 2.1. When considering it as an automaton on infinite words, it is universal as it accepts the word a^ω. On the other hand, the automaton is not universal when operating on finite words. It is easy to see that the language accepted by the automaton is $\{a^{2n} \mid n \geq 0\}$. Thus, for the acceptance condition defined above, *the universality problem for weighted automata on finite words is not equivalent to the universality problem for weighted automata on infinite words.* Moreover, at the end of Section 3.1, we discuss another acceptance condition under which the universality of a weighted automaton on finite words does not imply the universality of a weighted automaton on infinite words.

Counter machines A *Minsky machine*, introduced in [120] by Marvin Minsky, is a simple computation model that is crucial in proofs of Chapter 5. A deterministic two-counter

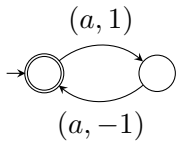


Figure 2.1: A weighted automaton over the unary alphabet, $\Sigma = \{a\}$, which is universal over infinite words but is not universal over finite words.

Minsky machine (2CM) is a tuple (Q, T, s_0), where Q is the finite set of states and

$$T \subseteq Q \times \{c_{i}{+}{+}, c_{i}{-}{-}, c_{i}{=}{=}0 \mid i = 1, 2\} \times Q$$

is the finite set of labelled transitions to increment, decrement or test for zero one of the counters, and s_0 is the initial state. In a deterministic two-counter Minsky machine, the set Q contains a special sink state $\bot$, such that there is no outgoing transition from $\bot$. Moreover, from all $s \in Q \setminus \{\bot\}$, either there is only one outgoing transition with the label $c_1{+}{+}$ or $c_2{+}{+}$, or there are exactly two outgoing transitions with respective labels $c_1{-}{-}$ and $c_1{=}{=}0$, or $c_2{-}{-}$ and $c_2{=}{=}0$. A *configuration* of a 2CM is a pair $[s, (y, z)] \in Q \times (\mathbb{Z}^+)^2$, representing a state and a pair of counter values. The *run* of a 2CM is a finite or infinite sequence of configurations that starts from $[s_0, (0, 0)]$ and follows the transitions of the machine incrementing and decrementing the counters according to the labels. As usual, a transition with a label $c_i{=}{=}0$ can only be taken when the counter i is zero and a transition with a label $c_i{-}{-}$ can only be taken when the counter i is positive.

Note that there is only one possible run in a deterministic two-counter Minsky machine. Indeed, when there are two outgoing transitions, only one of them can be executed, depending on the value of the counter that the transitions update or test for zero. The *halting problem* of 2CM is to decide, given a 2CM, whether the run reaches a configuration with state $\bot$, in other words whether the run halts. This problem is known to be undecidable for deterministic two-counter Minsky machines [120]. Another well-known undecidable problem for 2CM is whether the machine halts with both counters equal to zero. We are interested in a more general question: whether in the run of a 2CM both counters are zero at some point. This problem is undecidable and the proof follows from the halting problem by modifying a 2CM to ensure that both counters are zero only in the halting state; see [137] for a proof.

Theorem 2.4. *Let (Q, T, s_0) be a deterministic two-counter Minsky machine. It is undecidable whether in the run of (Q, T), a configuration in $Q \times \{(0,0)\} \setminus \{(s_0, (0,0))\}$ appears.*

We can assume that the first move of a 2CM is an increment of either c_1 or c_2. Indeed, otherwise the problem is trivial as the second configuration is in $Q \times \{(0,0)\}$.

Register machines with polynomial updates A *polynomial register machine* (PRM) is a tuple $\mathcal{R} = (S, \Delta)$, where S is the finite set of states, $\Delta \subseteq S \times \mathbb{Z}[x] \times S$ is the set of transitions labelled with *update polynomials*. A transition $\langle s, p(x), s' \rangle$ is often written as $s \xrightarrow{p(x)} s'$. A *configuration* c of $\mathcal{R}$ is a tuple $[s, z] \in S \times \mathbb{Z}$. A configuration $[s, z]$ is said to yield a configuration $[s', y]$ if there is a transition $\langle s, p(x), s' \rangle \in \Delta$ such that $p(z) = y$. This is denoted by $[s, z] \to_{\mathcal{R}} [s', y]$. The reflexive and transitive closure of $\to_{\mathcal{R}}$ is denoted by $\to^*_{\mathcal{R}}$. The *reachability problem* is to decide, given two configurations $[s_0, x_0]$ and $[s_f, x_f]$, whether $[s_0, x_0] \to^*_{\mathcal{R}} [s_f, x_f]$ holds. It is easy to reduce the general reachability problem to $[s_0, 0] \to^*_{\mathcal{R}'} [s_f, 0]$ for some $\mathcal{R}'$. Note, that when considering d-dimensional polynomial updates, the updates are applied componentwise, i.e., $p(x_1, \ldots, x_d) = (p_1(x_1), \ldots, p_d(x_d))$, where $p_i(x) \in \mathbb{Z}[x]$.

Linear-bounded automata A *linear-bounded automaton* (LBA) is a Turing machine with a tape bounded by a linear function of the length of the input. Equivalently, an LBA can be defined as a Turing machine with a finite tape. We denote an LBA $\mathcal{M}$ by a tuple (Q, Γ, δ), where Q is the finite set of states, $\Gamma = \{\triangleright, \triangleleft, 0, 1\}$ is the finite *tape alphabet*, containing two special letters $\triangleright$ and $\triangleleft$, which mark the left and right borders of the tape. The transition function δ is a mapping from $Q \times \Gamma$ to $Q \times \Gamma \times \{L, R\}$, where L and R tell the read/write head to move left or right, respectively. The automaton respects the boundary letters, that is, if $\delta(q_1, \triangleleft) = (q_2, b, D)$, then $b = \triangleleft$ and $D = L$, and symmetrically if $\delta(q_3, \triangleright) = (q_4, b', D')$, then $b' = \triangleright$ and $D' = R$, for any states $q_1, q_3 \in Q$ and where $q_2, q_4 \in Q$. A *configuration* is a triple $[q, i, \triangleright w \triangleleft]$, where $w \in \{0,1\}^n$ and $i = 0, \ldots, n+1$. Intuitively, in the configuration the automaton is in state q, the read/write head is in the ith cell and w is written on the tape. Let $\to^*_{\mathcal{M}}$ be the reflexive and transitive closure of the transition relation $\to_{\mathcal{M}}$ defined in the usual way. The *reachability problem* for a given LBA is to decide whether, for given $q_0, q_f \in Q$, $[q_0, 0, \triangleright 0^n \triangleleft] \to^*_{\mathcal{M}} [q_f, 0, \triangleright 0^n \triangleleft]$ holds and is a well-known PSPACE-complete problem. Without loss of generality, we can assume that q_f appears only in the configuration $[q_f, 0, \triangleright 0^n \triangleleft]$. Furthermore, we can enumerate the states

such that q_0 is the first state and q_f is the last state, i.e., $q_f = q_{|Q|-1}$.

Alternating pushdown systems A *pushdown system* (PDS) $\mathcal{P}$ is a triple (Q, Σ, Δ, c_0), where Q is the finite set of states, Σ is the finite alphabet, called *stack alphabet*, $\Delta \subseteq Q \times \Sigma \times Q \times \Sigma^*$ is the set of rewrite rules of the stack, and $c_0 \in Q \times \Sigma^*$ is the initial configuration. Intuitively, a rule $\langle q, a, q', u\rangle \in \Delta$ means that the system moves from a state q to q', popping a from the top of the stack and pushing u into the stack. A *configuration* of $\mathcal{P}$ is $[q, u] \in Q \times \Sigma^*$. A *run* is a sequence of configurations of $\mathcal{P}$, starting from an initial configuration $[q_0, u_0]$, respecting stack rewriting rules. That is, if $[q, ua]$ is followed by $[q', v]$, then there exists a rule $\langle q, a, q', v'\rangle$ such that $v = uv'$. The associated decision problem is whether a configuration from $Q \times \{\varepsilon\}$ appears in a run of $\mathcal{P}$. In a graphical presentation, a rule $\langle q, a, p, u\rangle$ is denoted by $q \xrightarrow{(a,u)} p$.

An alternating pushdown system [23, 144], is an extension of PDS, where the states are partitioned into two sets $Q_\exists$ and $Q_\forall$ and, intuitively, the system non-deterministically chooses a transition from a state in $Q_\exists$ and explores all transitions from a state in $Q_\forall$. That is, we are interested whether a configuration in $Q \times \{\varepsilon\}$ appears in such a branching run in all paths. The complexity of this decision problem was shown to be EXPTIME in [144].

2.3 Attacker-Defender games

Games *Attacker-Defender games* are two-player zero-sum games with perfect information. Starting from some initial configuration, each move of Adam (Defender) is followed by a move of Eve (Attacker). Eve aims to reach a target configuration, while Adam tries to keep Eve from reaching the target configuration. Eve has a winning strategy if she can eventually reach the target configuration regardless of Adam's moves. The main computational question is to check whether Eve has a winning strategy for a given set of moves, initial and target configuration.

More formally, a game is played on an arena X, which is a finite or infinite set of configurations, with two special elements — an initial configuration x_0 and a target configuration x_f. Eve has a set of moves E and Adam has a set of moves A such that, for each $w \in E \cup A$ and $x \in X$, $x \cdot w \in X$. Usually, the arena is clear from the context and we do not state it explicitly. For example, in d-dimensional robot games [9, 62, 125] and d-dimensional matrix games on vectors, the arena is $X = \mathbb{Z}^d$, while in weighted word games, the arena is $X = \mathrm{FG}(\Gamma) \times \mathbb{Z}$.

The game proceeds as follows: starting from x_0, first Adam chooses a move a from his set A and applies it to x_0 and then Eve chooses a move from her set E and applies it to $x_0 \cdot a$. By repeating this process, the players create an infinite sequence of elements of X called a *play*, $\pi = (x_0, x_0a_1, x_0a_1e_1, x_0a_1e_1a_2, x_0a_1e_1a_2e_2, \ldots)$, where each $a_i \in A$ and $e_j \in E$. A play π is *winning* for Eve if x_f appears at some configuration of the play. To make the special case, where the initial and target configurations are the same element, i.e., $x_0 = x_f$, nontrivial, we require that for a play π to be winning, x_f is not the first element of π. A *strategy* of Eve is a function $\sigma_E : X \to E$ that tells Eve which move to apply in each configuration. A strategy σ_A of Adam is defined analogously. We say that a play π is *consistent* with a strategy σ_E if, for two consecutive elements of π, x_i and x_{i+1}, where i is odd, it holds that $x_{i+1} = x_i \cdot \sigma_E(x_i)$. We define consistency with Adam's strategy σ_A symmetrically. A strategy σ_E (resp., σ_A) is said to be a *winning strategy* for Eve (resp., Adam), if all consistent plays are winning. As a consequence of [116], Attacker-Defender games are determined, that is, Adam has a winning strategy if Eve does not. See [75] for more details on games.

Note that the games are often considered on directed finite graphs with the vertices partitioned between two players, and the moves are edges of the graph. In most of our formulations, the graph has two vertices, one for Eve and one for Adam, no self-loops, and the edges from Eve's (resp., Adam's) vertex to Adam's (resp., Eve's) vertex are labelled with elements of E (resp., A).

Next, we look at particular games and fix the notation used when describing them.

Counter reachability games A d-dimensional counter reachability game (CRG) is a tuple (G, c_0) that consists of a directed graph $G = (V, F)$, where the set of vertices is partitioned into two parts, V_E and V_A, each edge $e \in F \subseteq V \times \mathbb{Z}^d \times V$ is labelled with vectors in $\mathbb{Z}^d$, and $c_0 \in V \times \mathbb{Z}^d$ is the initial configuration. We say that the vertex in the first component of an edge is the *source* vertex. A *degree* $\deg(v)$ of a vertex $v \in V$ is the number of edges with source v. That is, $\deg(v) = |\{(v, \mathbf{x}, v') \in F\}|$. A *configuration* of the game is $[v, \mathbf{x}]$, a successive configuration is $[v', \mathbf{x} + \mathbf{x}']$, where an edge $(v, \mathbf{x}', v') \in F$ is chosen by Eve if $v \in V_E$ or by Adam if $v \in V_A$. The goal of Eve is to reach the *final configuration* $[v_f, \mathbf{0}_d]$ for some $v_f \in V$ from the given initial configuration $c_0 = [v_0, \mathbf{x}_0]$, while the goal of Adam is to keep Eve from reaching $[v_f, \mathbf{0}_d]$. A *strategy* for a player is a function that maps a configuration to an edge that can be applied. We say that Eve has a *winning strategy* if she can reach the final configuration regardless of the strategies of

Adam. By the definition of determinacy, Adam has a winning strategy if Eve does not have a winning strategy. In the figures, we use ◯ for Eve's states and □ for Adam's states (diamonds represent arbitrary vertices). A two-dimensional counter reachability game is illustrated in Figure 2.2a.

$\mathbb{Z}^d$-VAS games A d-dimensional *game on integer vector addition system* ($\mathbb{Z}^d$-VAS game), introduced as robot game in [62], is a special case of the counter reachability games, where the graph consists of only two vertices, v_0 of Adam and v of Eve, and edges are of the form $(v_0, \mathbf{x}, v)$ and $(v, \mathbf{x}, v_0)$ (i.e., there are no self-loops). The goal of Eve is to reach the configuration $[v_0, \mathbf{0}_d]$. That is, a $\mathbb{Z}^d$-VAS game consists of two players, Eve and Adam, having a set of vectors E, A over $\mathbb{Z}^d$, respectively, and an *initial vector* $\mathbf{x}_0$. Starting from $\mathbf{x}_0$ players add a vector from their respective sets to the current configuration of the game in turns. As in counter reachability games, Eve tries to reach the origin while Adam tries to keep Eve from reaching the origin. The decision problem concerning $\mathbb{Z}^d$-VAS games is, for a given $\mathbb{Z}^d$-VAS game $(A, E, \mathbf{x}_0)$, to decide whether Eve has a winning strategy to reach $\mathbf{0}_d$ from $\mathbf{x}_0$. A two-dimensional integer vector addition system game is illustrated in Figure 2.2b.

In Section 4.3 and Section 5.3, we consider VAS games, where the configurations are non-negative, that is, the game is played under VAS semantics.

$\mathbb{Z}^d$-VASS games An extension of $\mathbb{Z}^d$-VAS games where players have control states is called *games on integer vector addition system with states* ($\mathbb{Z}^d$-VASS games). A d-dimensional $\mathbb{Z}^d$-VASS consists of (A, E, c_0), where A is the finite subset of $Q_A \times \mathbb{Z}^d \times Q_A$ that Adam can apply during his turn and E is the finite subset of $Q_E \times \mathbb{Z}^d \times Q_E$ of Eve, and $c_0 \in Q_E \times Q_A \times \mathbb{Z}^d$ is the initial configuration. A *configuration* is now a triple $[s, t, \mathbf{v}]$ consisting of Eve's control state s, Adam's control state t and the *counter vector* $\mathbf{v} \in \mathbb{Z}^d$. Eve updates her control state when she makes a move: in the configuration $[s, t, \mathbf{v}]$, for any vector $\mathbf{v}$, only moves of the form $\langle\!\langle s, \mathbf{x}, s' \rangle\!\rangle$ are enabled, and with one such move the new configuration is $[s', t, \mathbf{v} + \mathbf{x}]$. Adam updates his control state when he makes a move in a similar fashion. Eve *wins* if, and only if, after her turn, the configuration is $[s, t, \mathbf{0}_d]$ for any $s \in Q_E$ and any $t \in Q_A$. In the decision problem associated with $\mathbb{Z}^d$-VASS games, we are asked whether Eve has a winning strategy from the given initial configuration.

Remark 2.5. Note that in [1, 29, 138], where VASS games are considered, the game is played on a directed graph with vertices partitioned between the two players, as in counter

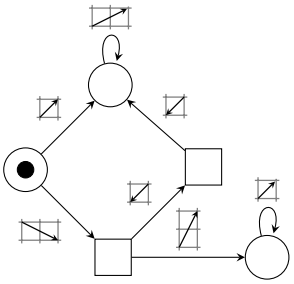
reachability games. This definition is not convenient for the k-control games we define later on. In Chapter 4, we show that these two definitions are equivalent for one-dimensional games. The result extends in a natural way to d-dimensional games.

In order to indicate whose turn it is in the configuration $[s, t, \mathbf{v}]$, we put a dot above s if it is Eve's turn, or above t if it is Adam's turn. That is, the respective configurations are $[\dot{s}, t, \mathbf{v}]$ and $[s, \dot{t}, \mathbf{v}]$. In the figures, the dot is placed inside the state (e.g., $\dot{\boxdot}$ if it is Adam's turn). When depicting stateless games, we use colours to distinguish players' moves. Adam's moves are in green and Eve's moves are in blue. A d-dimensional game on an integer vector addition system with states is illustrated in Figure 2.2c.

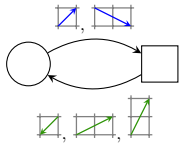
We define plays and strategies in terms of $\mathbb{Z}^d$-VASS games. A *play* is an infinite sequence of configurations of a game, $\pi = c_0 c_1 c_2 \cdots$, where if $c_i = [\dot{s}, t, \mathbf{x}]$, then $c_{i+1} = [s', \dot{t}, \mathbf{x}']$ if $\langle\!\langle s, \mathbf{x}' - \mathbf{x}, s' \rangle\!\rangle \in E$, or if $c_i = [s, \dot{t}, \mathbf{x}]$, then $c_{i+1} = [\dot{s}, t', \mathbf{x}']$ if $\langle\!\langle t, \mathbf{x}' - \mathbf{x}, t' \rangle\!\rangle \in A$. A play is *winning* for Eve, if the target configuration appears in it. On the other hand, if the target configuration never appears in the play, then the play is winning for Adam. A *strategy* of Eve is a function $\sigma_E : Q_E \times Q_A \times \mathbb{Z}^d \to E$ that tells Eve which move to use in the current configuration. We define Adam's strategy σ_A symmetrically. We say that a play π is *consistent* with a strategy σ_E if for configurations $c_i = [\dot{s}, t, \mathbf{x}]$ and $c_{i+1} = [s', \dot{t}, \mathbf{x}']$, it holds $\sigma_E([s, t, \mathbf{x}]) = \langle\!\langle s, \mathbf{x}' - \mathbf{x}, s' \rangle\!\rangle$. We define consistency with Adam's strategy σ_A symmetrically. A strategy σ_E (resp., σ_A) is a *winning strategy*, if all consistent plays are winning for Eve (resp., Adam). As for Attacker-Defender games, the determinacy result of Martin [116] applies, and thus, Adam has a winning strategy if and only if Eve does not have a winning strategy.

Flat $\mathbb{Z}$-VASS games The *flat* $\mathbb{Z}$-VASS games are a subclass of the $\mathbb{Z}$-VASS games where Eve is stateless, that is, all the moves of Eve are of the form $\langle\!\langle s, z, s \rangle\!\rangle$, and Adam's states are flat, i.e., without nested loops. In other words, we have an ordering of states of Adam $\{t_0, \ldots, t_k\}$ such that $\langle\!\langle t_i, z, t_j \rangle\!\rangle \in A$ only if $i \leq j$. Note that, unlike the usual definition of flat systems, we allow several self-loops for a state. A d-dimensional game on a flat integer vector addition system with states is illustrated in Figure 2.2d.

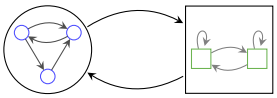
k-control $\mathbb{Z}$-VASS games A *k-control* $\mathbb{Z}^d$-VASS game is a variant of $\mathbb{Z}^d$-VASS games where one of the players can *skip* his or her turn and he or she can play at most k non-skipping moves during a play. We say that the k-control $\mathbb{Z}^d$-VASS game has a *safety* objective if Adam has k moves and *reachability* objective if Eve has k moves. That is, in safety of



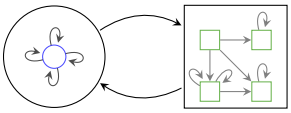
(a) A two-dimensional CRG.



(b) A $\mathbb{Z}^2$-VAS game.



(c) A $\mathbb{Z}^d$-VASS game.



(d) A flat $\mathbb{Z}^d$-VASS game.

Figure 2.2: Illustrations of four games.

k-control $\mathbb{Z}^d$-VASS game, the game now consists of a $\mathbb{Z}^d$-VASS game $(A, E, \mathbf{x}_0)$ together with a counter i that starts at 0 and is at most k. A strategy $\sigma_A : Q_E \times Q_A \times \mathbb{Z}^d \to A \cup \{\textsc{Skip}\}$ of Adam takes i into account and returns Skip if $i = k$, otherwise it either returns a move in A and increments i or returns Skip and does not increment i. The reachability variant is defined analogously.

2.4 $\mathbb{Z}$-VAS games, where each player has two moves

As an introductory example, consider a very limited fragment of $\mathbb{Z}^d$-VAS games, where both players have two moves. That is, let $E = \{\mathbf{e}_1, \mathbf{e}_2\} \subseteq \mathbb{Z}^d$ be the move set of Eve and $A = \{\mathbf{a}_1, \mathbf{a}_2\} \subseteq \mathbb{Z}^d$ be the move set of Adam. The initial vector is $\mathbf{x}_0 \in \mathbb{Z}^d$ and the origin is the target. In other words, we are interested whether by turn-based addition of vectors from A and E to $\mathbf{x}_0$, Eve can not only reach the origin, but to ensure that she can reach the origin regardless of the moves Adam plays.

In the general scenario, where the number of moves the players have is not fixed, there is a simple necessary and sufficient condition to determine whether the winning set of Eve is empty or not.

Lemma 2.6 (Arul, Reichert [9])**.** *The winning set in a* $\mathbb{Z}^d$-VAS *game* $(A, E, \mathbf{x}_0)$ *is nontrivial if and only if there exists a vector* $\mathbf{x} \neq (0, \ldots, 0)$ *such that, for all moves of Adam* $\mathbf{a} \in A$,

there exists a move $\mathbf{e} \in E$ *of Eve such that* $\mathbf{a} + \mathbf{e} = -\mathbf{x}$.

The claim is obvious as this is exactly what we mean when we say that Eve has a winning strategy to reach $(0, \ldots, 0)$ from $\mathbf{x}$ in one turn – regardless of which move Adam plays, Eve can play such a move that the origin is reached. We can iterate the lemma to construct bigger and bigger winning sets for Eve giving us a semi-algorithm, Algorithm 2.1, to solve $\mathbb{Z}^d$-VAS games. If there was a bound on the number of iterations, then we would have an algorithm to solve $\mathbb{Z}^d$-VAS games. Unfortunately, such a bound does not exist, as we will prove in Chapter 5 that already two-dimensional $\mathbb{Z}$-VAS games are undecidable. Even when $d = 1$, that is, when the game is played on integer line, finding the bound is not easy. In [9], the exponential bound was found by using nontrivial results on the Frobenius problem.

input : A $\mathbb{Z}^d$-VAS game $(A, E, \mathbf{x}_0)$.
output : The player that has a winning strategy in the game.

```
preX ← {(0, ..., 0)};
Moves ← {−a − e | a ∈ A, e ∈ E};
repeat
    X ← preX;
    if x_0 ∈ X then return Eve;
    foreach x ∈ X and y ∈ Moves do
        if forall a ∈ A exists e ∈ E such that x + y + a + e ∈ X then
            preX ← preX ∪ {x + y};
        end
    end
until X = preX;
if x_0 ∈ X then
    return Eve
else
    return Adam
end
```

Algorithm 2.1: A semi-algorithm solving a $\mathbb{Z}^d$-VAS game.

Consider then the game with two moves for each player. Assume that the winning set is not trivial, that is, for some $\mathbf{x} \in \mathbb{Z}^d$, $\mathbf{a}_1 + \mathbf{e}_1 = -\mathbf{x} = \mathbf{a}_2 + \mathbf{e}_2$ (up to renaming of the moves). We see that Eve has a winning strategy to reach the origin also from $2\mathbf{x}$. In fact, from $2\mathbf{x}$, Eve responds to moves of Adam in exactly the same way as she would from $\mathbf{x}$. Furthermore, we see that Eve has a winning strategy only from vectors of form $k\mathbf{x}$ for

some $k \in \mathbb{N}$. Indeed, from any other vector, either $\mathbf{a}_1 + \mathbf{e}_2$ or $\mathbf{a}_2 + \mathbf{e}_1$ does not result in a winning vector. The algorithm to decide which player has a winning strategy in $\mathbb{Z}^d$-VAS game $(A, E, \mathbf{x}_0)$, where $|A| = 2 = |E|$, is presented in Algorithm 2.2. Unfortunately, it does not scale even to games with three moves for each player.

Proposition 2.7. *Given a $\mathbb{Z}^d$-VAS game $(A, E, \mathbf{x}_0)$, where $|A| = |E| = 2$, it is decidable in linear time whether Eve has a winning strategy to reach $\mathbf{0}$ from $\mathbf{x}_0$.*

input : A $\mathbb{Z}^d$-VAS game $(A, E, \mathbf{x}_0)$, where $A = \{\mathbf{a}_1, \mathbf{a}_2\}$ and $E = \{\mathbf{e}_1, \mathbf{e}_2\}$.
output : The player that has a winning strategy in the game.

1 **if** $\mathbf{a}_1 + \mathbf{e}_1 = \mathbf{a}_2 + \mathbf{e}_2$ **and** $\mathbf{x}_0 = -k(\mathbf{a}_1 + \mathbf{e}_1)$ **then return** Eve;
2 **if** $\mathbf{a}_1 + \mathbf{e}_2 = \mathbf{a}_2 + \mathbf{e}_1$ **and** $\mathbf{x}_0 = -k(\mathbf{a}_1 + \mathbf{e}_2)$ **then return** Eve;
3 **return** Adam;

Algorithm 2.2: Solving a $\mathbb{Z}^d$-VAS game, where both players have two moves.

2.5 Properties of the particular subclass of the ωPCP

In the next chapter, we will prove that, in several Attacker-Defender games, it is undecidable which player has a winning strategy. The main ingredient of the proofs is the universality problem for integer weighted automata on infinite words. The undecidability of the universality problem is proved by a reduction from the ωPCP. In [82] and [83], a weighted automaton was constructed from an arbitrary instance of the ωPCP. In this section, we reiterate the construction of an instance of the ωPCP found in [80], highlighting the properties that simplify the construction of automata presented in the following chapter.

The ωPCP was shown to be undecidable for instances of size nine in [80]. The proof uses a reduction from the termination problem of the semi-Thue systems proved to be undecidable for the three-rule semi-Thue systems from [117]. Later, in [61], the ωPCP was proved to be undecidable for instances of size eight, using the same ideas but also utilising an encoding that helped to decrease the number of letters. We shall now present the construction from [80].

A *semi-Thue system* is a pair $\mathcal{T} = (\Sigma, R)$ consisting of an alphabet $\Sigma = \{a_1, \ldots, a_m\}$ and a relation set $R \subseteq \Sigma^* \times \Sigma^*$, the elements of which are called the *rules* of $\mathcal{T}$. For two words $u, v \in \Sigma^*$, we write $u \to_{\mathcal{T}} v$, if there are words u_1 and u_2 such that $u = u_1 x u_2$ and $v = u_1 y u_2$, where $(x, y) \in R$. Let $\to_{\mathcal{T}}^*$ be the reflexive and transitive closure of the

relation $\to_{\mathcal{T}}$. Therefore, we have $u \to_{\mathcal{T}}^{*} v$ if and only if either $u = v$ or there exists a finite sequence of words $u = v_1, v_2, \ldots, v_n = v$ such that $v_i \to_{\mathcal{T}} v_{i+1}$ for each $i = 1, 2, \ldots, n-1$. Let $w_0 \in \Sigma^*$ be a word, and $\mathcal{T} = (\Sigma, R)$ a semi-Thue system. If there does not exist any infinite sequence of words $w_1, w_2, \ldots$ such that $w_i \to_{\mathcal{T}} w_{i+1}$ for all $i \geq 0$, then we say that $\mathcal{T}$ *terminates* on w_0. Thus, $\mathcal{T}$ terminates on w_0 if all derivations starting from w_0 are of finite length. In the *termination problem*, we are given a word w_0 called an *input word*, and a semi-Thue system $\mathcal{T}$, and it is asked whether or not $\mathcal{T}$ terminates on w_0. As mentioned above the termination problem was proved to be undecidable for three-rule semi-Thue systems in [117].

First, let us show how we encode a semi-Thue system over an arbitrary sized alphabet into a semi-Thue system over a binary alphabet. Let $\mathcal{T}_1 = (\Sigma, R_1)$ be a semi-Thue system, where $\Sigma = \{a_1, a_2, \ldots, a_k\}$. Define a morphism $\varphi\colon \Sigma^* \to \{a, b\}^*$ with $\varphi(a_i) = ab^i a$ for all i. Then let $R_1' = \{(\varphi(u), \varphi(v)) \mid (u, v) \in R_1\}$ be a new set of rules, and define $\mathcal{T}_1' = (\{a, b\}, R_1')$. It is easy to see that $w \to_{\mathcal{T}_1} w'$ in $\mathcal{T}_1$ if and only if $\varphi(w) \to_{\mathcal{T}_1'} \varphi(w')$ in $\mathcal{T}_1'$. That is, $\mathcal{T}_1'$ and $\mathcal{T}_1$ are equivalent with respect to derivation. Therefore, if $\mathcal{T}_1$ has the undecidable termination problem, then so does the semi-Thue system $\mathcal{T}_1'$.

Let $\mathcal{T} = (\{a, b\}, R)$ be an n-rule semi-Thue system with the undecidable termination problem, and let the rules in $\mathcal{T}$ be $t_i = (u_i, v_i)$ for $i = 1, 2, \ldots, n$. We can assume that the rules t_i are encoded by φ. We are ready to define our instance of the infinite Post correspondence problem. The domain alphabet of the instance will be $\Sigma = \{a_1, a_2, b_1, b_2, d, \#\} \cup R$, where d is for the beginning and synchronisation and $\#$ is a special separator of the words in a derivation. Note that the rules in R are considered as letters in the alphabet. The image alphabet is $\{a, b, d, \#\}$. Let us define two special morphisms for $x \in \Sigma^+$. Morphisms l_x and r_x are called the *desynchronising* morphisms, and defined by $\ell_x(a) = xa$ and $r_x(a) = ax$ for each letter $a \in \Sigma$.

In [80], the following construction was given for a semi-Thue system $\mathcal{T}$ and an input word u: define the morphisms $g, h\colon \Sigma^* \to \{a, b, d, \#\}^*$ by (recall that for $t_i \in R$, we denoted $t_i = (u_i, v_i)$):

$$
\begin{array}{llll}
h(a_1) = dad, & g(a_1) = add, & h(b_1) = dbd, & g(b_1) = bdd, \\
h(a_2) = dda, & g(a_2) = add, & h(b_2) = ddb, & g(b_2) = bdd, \\
h(d) = \ell_{d^2}(u)dd\#d, & g(d) = dd, & h(\#) = dd\#d, & g(\#) = \#dd, \\
h(t_i) = d^{-1}\ell_{d^2}(v_i), & g(t_i) = r_{d^2}(u_i), \text{ for } t_i \in R. & &
\end{array}
\tag{2.2}
$$

In the image of t_i under h, the notation d^{-1} is a shorthand for the first letter of v_i being desynchronised by d rather than dd. In the special case, where $v_i = \varepsilon$, we define $h(t_i) = d$.

It was proved in [80] that each infinite solution of (g, h) is of the form

$$dw_1\#w_2\#w_3\#\cdots, \quad \text{where } w_j = x_j t_{i_j} y_j \tag{2.3}$$

for some $t_{i_j} \in R$, $x_j \in \{a_1, b_1\}^*$ and $y_j \in \{a_2, b_2\}^*$ for all j. Indeed, the image $g(w)$ is always of the form $r_{d^2}(v)$, and therefore, by the form of h, between two separators # there must occur exactly one letter $t \in R$. Also, the separator # must be followed by words in $\{a_1, b_1\}^*$ before the next occurrence of a letter $t \in R$. By the form of $h(t)$ the following words before the next separator must be in $\{a_2, b_2\}^*$. The form (2.3) follows when we observe that there must be infinitely many separators # in each infinite solution. Indeed, all solutions begin with a d, and there is one occurrence of # in $h(d)$ and no occurrences of # in $g(d)$. Later each occurrence of # is produced from # by both g and h. Therefore, there are infinitely many letters # in each infinite solution.

In the above construction from [80] the number of letters in the domain alphabet Σ is nine. It is possible to reduce the number of letters in the domain alphabet to eight following the transformation by Dong and Liu [61] where the letter b_2 is not needed anymore.

Example 2.8. Let $T = (\{a, b\}, \{t\})$ be a semi-Thue system with the input word $u = a$ and the rule $t = (a, aa)$. The corresponding instance of the ωPCP is

$$\begin{array}{llll}
h(a_1) = dad, & g(a_1) = add, & h(b_1) = dbd, & g(b_1) = bdd, \\
h(a_2) = dda, & g(a_2) = add, & h(b_2) = ddb, & g(b_2) = bdd, \\
h(d) = ddadd\#d, & g(d) = dd, & h(\#) = dd\#d, & g(\#) = \#dd, \\
h(t) = dadda, & g(t) = add. & &
\end{array}$$

In the following lemma, we show that in a solution, the image under g cannot be longer than the image under h.

Lemma 2.9. *Let (g, h) be as in (2.2), and let $p \in d\Sigma^*$ be such that $|g(p)| \geq |h(p)|$. Then $h(p) \not\leq g(p)$.*

Proof. The word $h(p)$ has more occurrences of # than $g(p)$. Indeed, $|h(p)|_\# = |g(p)|_\# + |p|_d \geq |g(p)|_\# + 1$, where $|\cdot|_\#$ is the number of letters # in the word. Therefore, $h(p) \not\leq g(p)$. □

The next lemma states that, in a word w beginning with the letter d, the first position where $h(w)$ and $g(w)$ differ (called the *error*) is reached in $h(w)$ at least one letter (of w) earlier than it is reached in $g(w)$.

Lemma 2.10. *Let (g,h) be as in (2.2) and assume that $w \in d\Sigma^\omega$ is not an infinite solution of the instance (g,h). Let $p = u'c$, where $c \in \Sigma$, be the shortest prefix of w such that $g(p) \not< h(p)$. Let r be the least position such that $h(p)[r] \neq g(p)[r]$. Then $r \leq |h(u')|$.*

Proof. Note first that $|p| \geq 2$ by the definition of $h(d)$ and $g(d)$. By the minimality of p, we have $g(u') \leq h(u')$.

Let v be the longest prefix of u' of the form in (2.3), that is,

$$v = dw_1\#w_2\#w_3\#\cdots w_n\#,$$

where

$$w_j = x_j t_{i_j} y_j$$

for some $t_{i_j} \in R, x_j \in \{a_1, b_1\}^*$ and $y_j \in \{a_2, b_2\}^*$ for all $j = 1, 2, \ldots, n$. Now

$$g(v) = v_1\#v_2\#\cdots v_n\#dd$$

and

$$h(v) = v_1\#v_2\#\cdots v_n\#ddv_{n+1}\#d,$$

where $v_i \in \{a, b, d\}^+$ for $i = 1, 2, \ldots, n+1$. More precisely,

$$\begin{aligned} v_1 &= dd\, g(w_1) = l_{d^2}(u)dd, \\ v_j &= dd\, g(w_j) = d\, h(w_{j-1})\, dd \quad \text{for } j = 2, \ldots, n, \quad \text{and} \\ v_{n+1} &= d\, h(w_n)\, dd. \end{aligned}$$

We prove that the error must appear within $v_{n+1}\#d$ in the image $h(v)$ which proves the claim. Assume to the contrary that the error is not within $v_{n+1}\#$, i.e., that the error appears after the last occurrence of # in the image $h(v)$. To cover $v_{n+1}\#d$ there must exist w_{n+1} such that $w_{n+1} = x_{n+1}t_{i_{n+1}}y_{n+1}$ (where $t_{i_{n+1}} \in R$, $x_{n+1} \in \{a_1, b_1\}^*$ and $y_{n+1} \in \{a_2, b_2\}^*$).

By the maximality of v, $vw_{n+1}\#$ is not a prefix of u', and therefore, $u' = vw_{n+1}$ and $c = \#$. But then $g(p) \leq h(p)$; a contradiction. $\square$

Chapter 3

Attacker-Defender games

In this chapter, we consider some Attacker-Defender games and show that it is undecidable which player has a winning strategy.

In particular, we introduce *matrix games on vectors*, where we show that, if both players are stateless and the moves correspond to very restricted linear transformations from $\mathrm{SL}(4, \mathbb{Z})$, the existence of a winning strategy is undecidable. To prove the undecidability in four-dimensional games, we first show undecidability of *word games*, where players are given words over a group alphabet and in an alternating way concatenate their words with the goal for Eve to reach the empty word. The games on words, over semigroup alphabets, are commonly used to prove results in language theory [101, 102, 113]. We, on the other hand, define word games over group alphabets.

Later, we show that it is possible to stretch the application of the proposed techniques to other models and frameworks. For example, we consider games on braids, which were recently studied in [26, 34]. Braids are classical topological objects that attracted a lot of attention due to their connections to topological knots and links, as well as their applications to polymer chemistry, molecular biology, cryptography, quantum computations and robotics [51, 59, 66, 72, 129]. In this chapter, we consider games on braids with only three or five strands, where the braid is modified by a composition of braids from a finite set with the target for Eve to reach the trivial braid. We show that it is undecidable to check for the existence of a winning strategy for three strands from a given nontrivial braid and for five strands starting from the trivial braid. The reachability with a single-player (i.e., with nondeterministic composition from a single set) was shown to be decidable for braids with three strands in [133].

The undecidability results of this chapter are proved by using a new language-theoretic result showing that the universality problem for weighted automata $\mathcal{A}$ on infinite words is undecidable. The acceptance of an infinite word w intuitively means that there exists a finite prefix p of w such that for the word p there is a path in $\mathcal{A}$ that has zero weight. From an instance of the infinite Post correspondence problem we construct the automaton $\mathcal{A}$ that accepts all infinite words if and only if the instance does not have a solution. As the infinite Post correspondence is undecidable [141], so is the universality problem for weighted automata on infinite words.

Please note that while the universality for weighted automata on finite words implies universality for weighted automata on infinite words, the statement does not hold the other way around. Therefore, the universality problem for weighted automata on infinite words is not equivalent to the universality problem for weighted automata on finite words. Our undecidability proof for weighted automata on infinite words follows the initial idea from [79] for mapping computations on words into a weighted (one-counter) automata model, which is extended with new constructions and formal proofs.

3.1 The universality problem for weighted automata on infinite words

In this section, we prove that the universality problem is undecidable for integer weighted automata on infinite words by reducing the instances (of the complement) of the infinite Post correspondence problem to the universality problem.

Let (g, h) be a fixed instance of the ωPCP as in Section 2.5. That is, $g, h\colon \Sigma^* \to B^*$, where $\Sigma = \{a_1, a_2, \ldots, a_m\}$ and $B = \{b_1, b_2, \ldots, b_{s-1}\}$. In our encoding of the ωPCP, we consider the letters of the alphabet B as natural numbers from 1 to $s-1$. We construct a nondeterministic integer weighted automaton $\mathcal{A} = (Q, \Sigma, \sigma, q_1, \{q_3\}, \mathbb{Z})$, where $Q = \{q_1, q_2, q_3\}$, corresponding to the instance (g, h) such that an infinite word $w \in \Sigma^\omega$ is accepted by $\mathcal{A}$ if and only if for some finite prefix p of w, $g(p) \not\preccurlyeq h(p)$ and $h(p) \not\preccurlyeq g(p)$. That is, only the solutions of the ωPCP will be rejected by $\mathcal{A}$ and if all infinite words are accepted by $\mathcal{A}$ (i.e., it is universal) then the given instance of the ωPCP does not have a solution.

Note that our automaton is complete, i.e., there is a transition labelled with (a, z) from each state q_i for every $a \in \Sigma$ and some $z \in \mathbb{Z}$.

In many existing reductions of the PCP or the ωPCP (for given morphisms g and h),

to show undecidability of other problems, it is required to explicitly construct and store images under g and h during the simulation of the PCP or the ωPCP. In our proof, we are only going to store some partial information about the difference between the lengths of the images under morphisms g and h and some finite information that will allow us to use the non-determinism of the automaton to guess and recognise a mismatch at some position of the images.

In order to present the idea of the proof, we will first illustrate the encoding of the ωPCP into the universality problem for a weighted automaton with two weights. One weight will be used for storing the distances between different positions of letters in images under g and h, and another one to store information about a particular letter encoded as an integer. The two weights are used only to make the intuition clearer and they will be merged into a single weight by storing the second weights in the least significant digits of the first one as the second weight is only used to store finitely many values.

Let us consider an instance of the ωPCP where the image under h is always longer than the image under g, and assume that the automaton reads $w \in \Sigma^\omega$. In the encoding of the ωPCP into a weighted automaton, we separate the run of the weighted automaton into five parts (A,B,C,D and E).

In part A, the automaton reads some finite prefix u of w for which it assumes that there are no errors in the images of letters of u under morphisms g and h. As there are no errors, i.e., $g(u) \leq h(u)$, it is not necessary to store the information on what the actual images are. Instead the automaton, for each letter a of u, only adds the differences of lengths of the images $h(a)$ and $g(a)$ to the first weight. In part B, the automaton reads the next letter of w, x, and guesses that an error will occur at the position k in the image of $h(x)$. The automaton adds to the first weight the difference of k and the length of the image of x under g and also adds $j_k = h(x)[k]$ to the second weight. Now the value of the first weight corresponds to the number of letters in the image of g between positions $|g(ux)|$ and $|h(u)| + k$; see Figure 3.1 for an illustration. From now on, the rest of the image of $h(w)$ starting from the position $|h(u)| + k + 1$ will not affect the acceptance of the word w. For the word w to be accepted, the letter in position $|h(u)| + k$ of $h(w)$ needs to be different from the letter at the same position of $g(w)$. This happens when both weights are zero. Zero in the first weight guarantees that we consider letters of both $h(w)$ and $g(w)$ in the position $|h(u)| + k$. On the other hand, zero in the second weight guarantees that the two letters in the considered position are different. Next, in parts C and D, the automaton will check whether both conditions can be met.

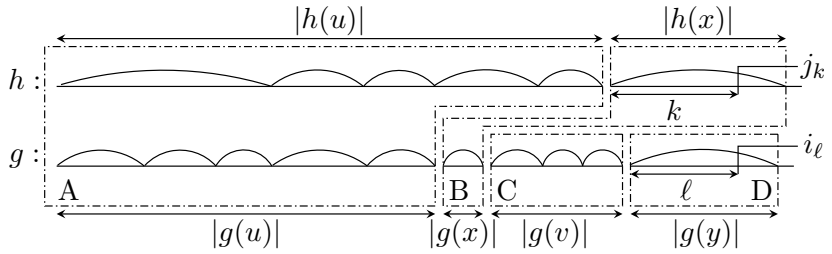


Figure 3.1: An illustration of a computation of the ωPCP highlighting the first four parts, A, B, C and D. Part E is not depicted.

In part C, the automaton continues reading w by reading v, assuming that the letter corresponding to the erroneous position under h has not been read yet. For each letter a of v read, the length of the image $g(a)$ is subtracted from the first weight. Finally, in part D, the automaton guesses that reading the next letter, y, the error will occur in the image of $g(y)$ at the position ℓ. That is, the automaton subtracts ℓ from the first weight and from the second weight, some letter $i_\ell \neq g(y)[\ell]$, i.e., any letter that is not in the ℓth position in the image of y under g. In the final part, E, the automaton is in the final state where the weights are no longer modified and the rest of the word w is read.

From the above description, in part E, the first weight is zero if the lengths of the images under h in part A, together with the position k, equal the lengths of the images under g in parts A, B and C together with the position ℓ. That is, $|h(u)| + k = |g(uxv)| + \ell$. In other words, the automaton checked whether an error occurred in the same position of $h(w)$ and $g(w)$. On the other hand, the second weight can be zero only if the letter in the image under h differs from the letter in the image under g. In other words, both weights are zero if the automaton made a correct guess that $h(w)[i] \neq g(w)[i]$ for some $i \in \mathbb{Z}^+$. The whole process is illustrated in Figure 3.2.

As mentioned previously, we can merge the two weights into a single one. The second weight is bounded and is modified only twice (once to store a letter and once to compare). By multiplying the integers stored in the first weight by s (where s is larger than the size of the image alphabet B), we create enough space to store the second weight within the first one.

In the above description, we considered the case where the images under h were always longer than the images under g. Obviously, this is a strong assumption that does not hold for arbitrary instances of the ωPCP. Luckily for us, for the instances defined by (2.2),

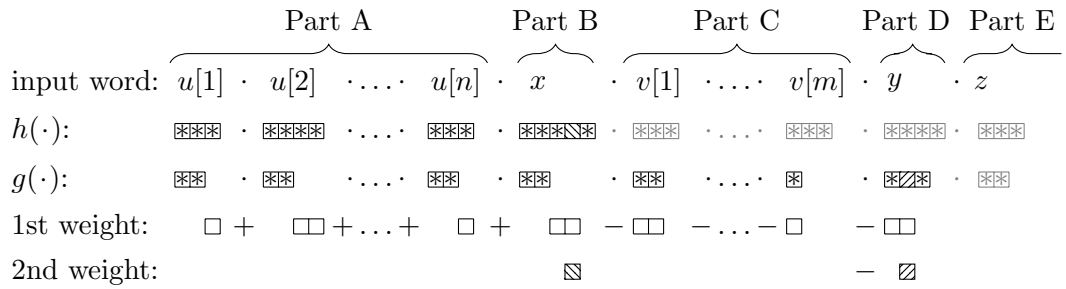


Figure 3.2: An illustration of a computation of the weighted automaton corresponding to an instance of the ωPCP. Here, ⊠ represents any letter of the image alphabet, while ⧅ is the letter $h(x)[k]$ and ▨ is the letter $g(y)[\ell]$.

by Lemma 2.9, the converse implies that the word is not a solution and furthermore by Lemma 2.10, that the first error appeared while the image under h is longer than the image under g. It is also possible to construct a weighted automaton from an arbitrary instance of the ωPCP. In that case, we would need to have additional states to make sure that all possible non-solutions of the instance can be covered. In fact, in [82, 83], the more general weighted automaton is constructed from an arbitrary instance of the infinite PCP.

Now we are ready to formally define our weighted automaton with a single weight. Let us begin with the transitions of $\mathcal{A} = (\{q_1, q_2, q_3\}, \Sigma, \sigma, q_1, \{q_3\}, \mathbb{Z})$, where $\Sigma = \{a_1, \ldots, a_{m-1}, d\}$. The automaton is depicted in Figure 3.3. Recall that the size of the image alphabet B is $s - 1$. Let us define the transitions of the automaton. First, for each $a \in \Sigma$, let

$$\langle q_1, a, q_1, s \cdot (|h(a)| - |g(a)|)\rangle\,, \quad \langle q_2, a, q_2, s \cdot (-|g(a)|)\rangle\,, \quad \langle q_3, a, q_3, 0\rangle$$

be in σ. For error checking we need two types of transitions – first to guess that the error has occurred in the image and then to verify the error. Let $h(a) = b_{j_1} b_{j_2} \cdots b_{j_{n_1}}$, where $b_{j_k} \in B$, for each index $1 \leq k \leq n_1$, and $g(a) = b_{i_1} b_{i_2} \cdots b_{i_{n_2}}$, where $b_{i_\ell} \in B$. Then let, for each $k = 1, \ldots, n_1$, (i.e., $j_k \in \{1, \ldots, s-1\}$ for all $k = 1, \ldots, n_1$),

$$\langle q_1, a, q_2, s \cdot (k - |g(a)|) + j_k\rangle \in \sigma. \tag{3.1}$$

For each $\ell = 1, \ldots, n_2$ and for each letter $b_c \in B$ such that $b_{i_\ell} \neq b_c \in B$, let

$$\langle q_2, a, q_3, -s\ell - c\rangle \in \sigma. \tag{3.2}$$

We call the transitions in (3.1), the *error guessing transitions* and in (3.2), the *error verifying transitions*. Finally, for all $b \in \Sigma \setminus \{d\}$, let $\langle q_1, b, q_3, 0\rangle \in \sigma$.

Intuitively, the self-loops in state q_1 correspond to part A, error guessing transitions (3.1) to part B, self-loops in state q_2 to part C, error verifying transitions (3.2) to part D, and self-loops in state q_3 to part E. While the transition described last, that is, $\langle q_1, b, q_3, 0\rangle$, correspond to the word starting by a letter other than d.

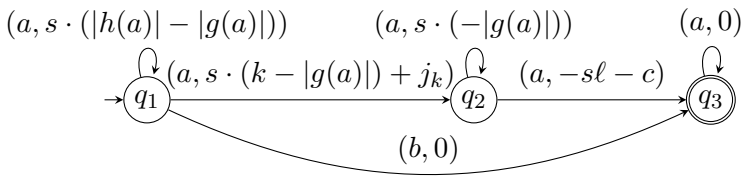


Figure 3.3: The weighted automaton $\mathcal{A}$. In the figure, $a \in \Sigma$ and $b \in \Sigma \setminus \{d\}$.

The idea is to keep track of differences in lengths of images under g and h multiplied by s, and then to guess and verify an error in the images by storing letters of the image alphabet in the least significant digits of the integer weight. By Lemma 2.10, if an error occurs, the difference in lengths of images is large enough that the image of the second morphism has to catch up before the error can be verified.

The following lemma shows that for each non-solution of the ωPCP, there exists a computation path with zero weight ending in the state q_3.

Lemma 3.1. *Let $w \in \Sigma^\omega$. Then w is a solution of an instance (g, h) of the form (2.2) of the ωPCP if and only if $w \notin L(\mathcal{A})$.*

Proof. Let $w = c_1 c_2 \cdots$ with $c_i \in \Sigma$ for all $i = 1, 2, \ldots$. Assume first that w is not a solution of the instance (g, h) of the form (2.2). Now either the first letter is d or not. In the latter case w is accepted by a path starting with $\langle q_1, c_1, q_3, 0\rangle$.

In the first case, by Lemma 2.9 and Lemma 2.10, there exists a prefix p of w such that $g(p) \not\leq h(p)$ and the first error position is reached in the image of $h(w)$ at least one letter (of w) before it is reached in the image of $g(w)$. Let r be the minimal position for which $h(w)[r] \neq g(w)[r]$. In other words, for $p = dc_2 \cdots c_n$, there exists a position $t < n$ such that

$$r = |h(dc_2 \cdots c_{t-1})| + k, \text{ where } k \leq |h(c_t)|, \text{ and}$$
$$r = |g(dc_2 \cdots c_{n-1})| + \ell, \text{ where } \ell \leq |g(c_n)|.$$

Denote $h(w)[r] = b_{j_k}$. It is the kth letter of the image $h(c_t)$, and $g(w)[r]$ is the ℓth letter of the image $g(c_n)$. By the choice of r, these letters are different.

Now, w is accepted in the state q_3 with the following path: First, the prefix $dc_2 \cdots c_{t-1}$ is read in the state q_1 with weight

$$s \cdot (|h(dc_2 \cdots c_{t-1})| - |g(dc_2 \cdots c_{t-1})|).$$

Next, when reading c_t, the error guessing transition $\langle q_1, c_t, q_2, s \cdot (k - |g(c_t)|) + j_k \rangle$ is taken and then the word $c_{t+1} \cdots c_{n-1}$ is read in the state q_2 with weight

$$s \cdot (-|g(c_{t+1} \cdots c_{n-1})|).$$

Finally, while reading the letter c_n, the state q_3 is reached by the error verifying edge $\langle q_2, c_n, q_3, -s\ell - j_k \rangle$. Note that such an error verifying edge exists as the ℓth letter in $g(c_n)$ is not equal to the kth letter, b_{j_k}, of $h(c_t)$. Naturally, after reaching q_3 the weight does not change as for all letters there are only transitions with zero weight. Now the weight of the above path is

$$\begin{aligned}
\gamma(p) &= s \cdot (|h(dc_2 \cdots c_{t-1})| - |g(dc_2 \cdots c_{t-1})|) + s \cdot (k - |g(c_t)|) + j_k \\
&\quad + s \cdot (-|g(c_{t+1} \cdots c_{n-1})|) - s\ell - j_k \\
&= s \cdot \big(|h(dc_2 \cdots c_{t-1})| + k - |g(dc_2 \cdots c_{n-1})| - \ell\big) \\
&= s \cdot (r - r) = 0.
\end{aligned}$$

Therefore, w is accepted, as claimed.

Next, we prove that if w is a solution of the instance (g, h), then it is not accepted by $\mathcal{A}$. Assume contrary to the claim that w is a solution and there is an accepting path of w in $\mathcal{A}$. As stated in (2.3), we have $w = dw_1 \# w_2 \# w_3 \# \cdots$, where $w_j = x_j t_{i_j} y_j$ for some $t_{i_j} \in R, x_j \in \{a_1, b_1\}^*$ and $y_j \in \{a_2, b_2\}^*$ for all j.

There are two possible computation paths for w. It can be accepted by a path visiting q_2 or not. In the second case, the prefix of w that is read in q_1 has to have equal lengths under g and h. By Lemma 2.9, w is not a solution of the instance of the ωPCP.

If the computation path visits q_2, then we can partition w into different parts according to the state-transition of the automaton. That is, w has a prefix $p = uxvy$, where $x, y \in \Sigma$, such that u is read in the state q_1 and v in the state q_2, and when reading the letter x the

path moves to q_2 and when reading the letter y the path moves to q_3. The weight $\gamma(p)$ of p is now

$$\begin{aligned}\gamma(p) &= s \cdot (|h(u)| - |g(u)|) + s \cdot (k - |g(x)|) + j_k + s \cdot (-|g(v)|) + (-s\ell - c) \\ &= s \cdot \big(|h(u)| + k - |g(uxv)| - \ell\big) + j_k - c,\end{aligned}$$

where $h(x)[k] = b_{j_k}$ and $g(y)[\ell] \neq b_c$. As $j_k < s$ and $c < s$, we have that $\gamma(p) = 0$ if and only if $|h(u)| + k = |g(uxv)| + \ell$ and $j_k = c$. Denote $r = |h(u)| + k$. Now, $\gamma(p) = 0$ if and only if $h(w)[r] = b_{j_k} \neq b_c = g(w)[r]$, which is a contradiction since w was assumed to be a solution of (g, h). □

We are ready to prove the main theorem of the section.

Theorem 3.2. *It is undecidable whether or not $L(\mathcal{A}) = \Sigma^\omega$ holds for a given three-state integer weighted automaton $\mathcal{A}$ on infinite words over alphabet Σ.*

Proof. Let $\mathcal{A}$ be the weighted automaton constructed in this section. The claim follows from Lemma 3.1 and the undecidability of the infinite PCP [80]. □

Corollary 3.3. *It is undecidable whether or not for a weighted automaton $\mathcal{A}$ on infinite words, there exists a word $w \in \Sigma^\omega$ such that, for every computation path π of w, all configurations $[q, u, z] \in \mathcal{R}(\pi)$ in a final state do not have zero weight.*

Proof. The statement formulates the condition for non-universality of weighted automata on infinite words. By the previous theorem, the universality problem is undecidable, and thus, so is its complement problem. □

Note that the number of the letters in the alphabet Σ in Theorem 3.2 is small. Indeed, $|\Sigma| = 9$ by the construction in (2.2). In contrast, the number of transitions is huge. The number of error guessing and verifying transitions depend on the lengths of the images. One of the rules of the three-rule semi-Thue system consists of encodings of all the rules of the 83-rule semi-Thue system that has an undecidable termination problem. Its image is several hundreds of thousands of letters long. It would be interesting to construct a weighted automaton from a different instance of the ωPCP with larger domain alphabet Σ, but shorter images. For example, in [85], the authors proved that the termination problem is undecidable for a semi-Thue system with 24 rules, where the length of each rule is at

most five. Constructing an instance of the infinite PCP from it using the technique of [80], could result in a weighted automaton with few transitions.

Example 3.4. Consider the ωPCP of Example 2.8. We will not present the whole weighted automaton as even for this small semi-Thue system and an ωPCP instance, the automaton has 112 transitions. Let a_1 be encoded as 1 and # as 4.

Let $w \in da_1a_1\Sigma^\omega$. It is easy to see that this is not a solution of the ωPCP. Indeed, $g(da_1a_1) = ddadd\mathbf{a}dd \not< ddadd\#ddaddad = h(da_1a_1)$. First, we show that guessing that the error is in the image of the first a_1 does not lead to an accepting computation path, since $g(da_1) = ddadd < ddadd\#ddad = h(da_1)$. Then we show that there is an accepting computation path when we guess that the error is in the image of the second a_1.

If we guess that the error will occur in the third position of the images, we need to store letter a and position three when reading d. This is done by using the transition

$$\langle q_1, d, q_2, s \cdot (k - |g(d)|) + j_k \rangle = \langle q_1, d, q_2, 5 \cdot (3-2) + 1 \rangle = \langle q_2, d, q_2, 6 \rangle .$$

Then we have to verify the error using a transition

$$\langle q_2, a_1, q_3, -s\ell - c \rangle = \langle q_2, a_1, q_3, -5 - c \rangle ,$$

where $c = 2, 3, 4$. After these two transitions the weight is at most -1 and thus w is not accepted with this path.

On the other hand, if we guess that the error will occur in the sixth position of the image, we use the transition

$$\langle q_1, d, q_2, 5 \cdot (6-2) + 4 \rangle = \langle q_1, d, q_2, 24 \rangle .$$

Then the first a_1 is read in the state q_2 with the transition $\langle q_2, a_1, q_2, -5 \cdot 3 \rangle$, after which the weight is 9. Then we verify the error with the transition $\langle q_2, a_1, q_3, -5 - 4 \rangle$. After these three transitions, the weight is 0 and the computation path has reached the state q_3. Thus, w is accepted by the automaton.

It is also natural to consider the emptiness problem for weighted automata on infinite words. That is, whether, for a given weighted automaton $\mathcal{A}$, $L(\mathcal{A}) = \emptyset$. In contrast to the result of Theorem 3.2, the emptiness problem is decidable.

Theorem 3.5. *It is decidable whether or not $L(\mathcal{A}) = \emptyset$ holds for an integer weighted automaton $\mathcal{A}$ on infinite words over alphabet Σ.*

Proof. Let $\mathcal{A}$ be a weighted automaton on infinite words. Consider it as a weighted automaton on finite words, $\mathcal{B}$, defined in [79]. Clearly, $L(\mathcal{A}) = \emptyset$ if and only if $L(\mathcal{B}) = \emptyset$. Indeed, an infinite word w is accepted by $\mathcal{A}$ if and only if there is a finite prefix u of w with $\gamma(u) = 0$. This u is accepted by $\mathcal{B}$. On the other hand, if some finite word u is accepted by $\mathcal{B}$ then an infinite word starting with u is accepted by $\mathcal{A}$.

In [78], it was shown that languages defined by weighted automata on finite words are context-free languages. It is well-known that emptiness is decidable for context-free languages (see for example [139]). □

Next, we list some straightforward corollaries. They all follow from Theorem 3.2 and are listed for completeness sake.

Corollary 3.6. *For weighted automata $\mathcal{A}$ and $\mathcal{B}$ on infinite words, the following language problems are undecidable:*

(i) Equality: Whether $L(\mathcal{A}) = L(\mathcal{B})$.

(ii) Inclusion: Whether $L(\mathcal{B}) \subset L(\mathcal{A})$.

(iii) Union: Whether $L(\mathcal{A}) \cup L(\mathcal{B}) = \Sigma^\omega$.

(iv) Regularity: Whether $L(\mathcal{A}) = R$, where R is an ω-regular language.

Proof. Let $\mathcal{A}$ be the automaton of Theorem 3.2.

(i) Let $\mathcal{B}$ be a weighted automaton on infinite words with one state q and transitions $\langle q, a, q, 0\rangle$ for all $a \in \Sigma$. Clearly, $L(\mathcal{B}) = \Sigma^\omega$. Now the automata $\mathcal{A}$ and $\mathcal{B}$ accept the same language if and only if $\mathcal{A}$ accepts Σ^ω.

(ii) Let $\mathcal{B} = (\{q_0, q_1\}, \Sigma, \sigma, q_0, \{q_1\}, \mathbb{Z})$, where

$$\sigma = \{\langle q_0, d, q_1, 0\rangle, \langle q_1, d, q_1, 1\rangle\} \cup \{\langle q_0, a, q_1, 1\rangle, \langle q_1, a, q_1, 0\rangle \mid a \in \Sigma \setminus \{d\}\}.$$

Let w be a solution of an instance (g, h) of the ωPCP. It is accepted by $\mathcal{B}$ but not by $\mathcal{A}$. Now if the instance (g, h) of ωPCP has a solution, then $L(\mathcal{B}) \not\subset L(\mathcal{A})$. On the other hand, if the instance (g, h) of ωPCP does not have a solution, then $L(\mathcal{B}) \subset L(\mathcal{A})$.

(iii) Let $\mathcal{B}$ be an automaton accepting the empty language. Now $L(\mathcal{A}) \cup L(\mathcal{B}) = \Sigma^\omega$ holds if and only if $L(\mathcal{A}) = \Sigma^\omega$.

(iv) The claim follows as Σ^ω is an ω-regular language. □

The previous statements were language-theoretic in nature, in the next corollary, we present a different undecidability result for weighted automata on infinite words.

Corollary 3.7. *It is undecidable whether $L(\mathcal{A}) = L(\mathcal{A}')$ for two weighted automata on infinite words, $\mathcal{A}$ and $\mathcal{A}'$, such that there exists a bijective mapping from edges of $\mathcal{A}$ to edges of $\mathcal{A}'$.*

Proof. Let $\mathcal{A}$ be the automaton of Theorem 3.2. Consider an automaton $\mathcal{A}'$ with a single state q'. Let $T_a = \{\langle q, a, p, z\rangle \in \sigma\}$ be the set of all transitions in $\mathcal{A}$ reading a letter a. Denote by n_a the size of the set T_a. Now, in $\mathcal{A}'$ for each letter a we add a transition $\langle q, a, q, -i\rangle$, where $i = 0, \ldots, n_a - 1$. Clearly, $L(\mathcal{A}') = \Sigma^\omega$. There is an obvious bijection between transitions of $\mathcal{A}$ and $\mathcal{A}'$. Now the automata accept the same language if and only if $\mathcal{A}$ accepts Σ^ω. □

The result of Theorem 3.2 also holds for a more restricted acceptance condition, similar to that of Büchi automata, where an infinite word is accepted if and only if there is a computation path with an infinite number of prefixes with weight zero. This is clear from the construction of the automaton. If the state q_3 is reached with zero weight, then it remains unchanged when additional letters are read. For this acceptance condition the universality of a weighted automaton on finite words does not imply the universality for a weighted automaton on infinite words. However, for simplicity, we use the original acceptance condition in the proofs in Section 3.2.

3.2 Applications to Attacker-Defender games

In this section, we provide a number of applications for our new result on the universality of weighted automata on infinite words. We connect the idea of a one-weight computation with games on different mathematical objects such as words, matrices, vectors and braids.

Following the result for the weighted automata on infinite words (Theorem 3.2) we can now define a simple scenario of an undecidable infinite-state game that can also be applied to other game frameworks. Let $\mathcal{A}$ be a weighted automaton. In the game, Adam will play

an input word letter-by-letter and Eve will simulate a run on $\mathcal{A}$ using the letters provided by Adam. In other words, Eve has to verify whether the provided word is accepted by $\mathcal{A}$.

In the above framework, Adam will have a winning strategy if there is a solution for the infinite Post correspondence problem and Eve will have a winning strategy otherwise.

3.2.1 Weighted word games

Let us define the Attacker-Defender game on words, where the moves of Eve and Adam correspond to concatenations of words (over a free group alphabet). Later on, we shall see that this game allows us to prove nontrivial results for games with *low-dimensional* linear transformations or topological objects just by using an injective homomorphism (i.e., a monomorphism) to map words to other mathematical objects.

A *weighted word game* consists of two players, Eve and Adam having sets of words $\{e_1, \ldots, e_r\} \subseteq \mathrm{FG}(\Gamma)$ and $\{a_1, \ldots, a_s\} \subseteq \mathrm{FG}(\Gamma)$ respectively, where Γ is a finite group alphabet, and integers $x_{e_1}, \ldots, x_{e_r}, x_{a_1}, \ldots, x_{a_s}$ corresponding to each word. That is, the players' move sets are

$$E = \{\langle\!\langle e_1, x_{e_1} \rangle\!\rangle, \ldots, \langle\!\langle a_r, x_{a_r} \rangle\!\rangle\} \text{ and } A = \{\langle\!\langle a_1, x_{a_1} \rangle\!\rangle, \ldots, \langle\!\langle a_s, x_{a_s} \rangle\!\rangle\}.$$

An initial configuration is the pair $[w, 0]$, where $w \in \mathrm{FG}(\Gamma)$ and 0 is the initial value of the weight, and the target configuration of this game is the group identity, i.e., the empty word, with zero weight. A *configuration* of a game at time t is $[w_t, x]$, where $w_t \in \mathrm{FG}(\Gamma)$ and $x \in \mathbb{Z}$, that we call *a weight*. In each round of the game, both Adam and Eve concatenate their words and update the weight. Clearly, $w_t = w \cdot a_{i_1} \cdot e_{i_1} \cdot a_{i_2} \cdot e_{i_2} \cdot \ldots \cdot a_{i_t} \cdot e_{i_t}$ after t rounds of the game, where e_{i_j} and a_{i_j} are words from the sets E and A respectively, and the weight is $\sum_{j=1}^{t}(x_{a_{i_j}} + x_{e_{i_j}})$. As usual, the decision problem for the word game is to check whether there exists a winning strategy for Eve to reach the empty word with zero weight.

Before proving the main theorem, we consider an auxiliary result. In the next lemma, we modify the automaton of Theorem 3.2 in order to remove self-loops.

Lemma 3.8. *It is undecidable whether or not $L(\mathcal{B}) = \Sigma^\omega$ holds for six-state integer weighted automata on infinite words over alphabet Σ, without self-loops.*

Proof. Let $\mathcal{A} = (\{q_1, q_2, q_3\}, \Sigma, \sigma, q_1, \{q_3\}, \mathbb{Z})$ be the automaton of Theorem 3.2. We construct an automaton $\mathcal{B} = (Q_\mathcal{B}, \Sigma, \sigma', q_1, \{q_3\}, \mathbb{Z})$, where $Q_\mathcal{B} = \{q_1, q_2, q_3, q_4, q_5, q_6\}$. The

idea is to have two copies of $\mathcal{A}$ and, instead of self-loops, we switch between the copies. In $\mathcal{B}$, the set of transitions σ' is defined as follows:

$$\sigma' = \{\langle q_i, a, q_j, z\rangle, \langle q_{i+3}, a, q_{j+3}, z\rangle \mid \langle q_i, a, q_j, z\rangle \in \sigma, i \neq j\}$$
$$\cup \{\langle q_i, a, q_{i+3}, z\rangle, \langle q_{i+3}, a, q_i, z\rangle \mid \langle q_i, a, q_i, z\rangle \in \sigma\}.$$

The automaton is depicted in Figure 3.4. It is easy to see that both automata accept the same language. □

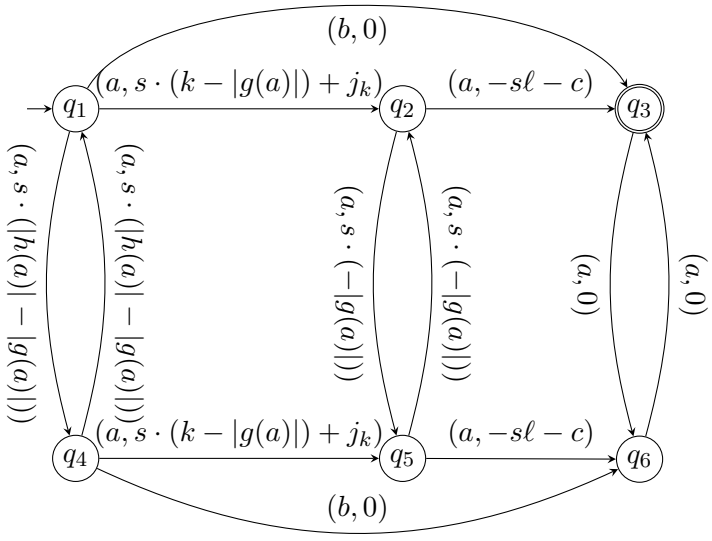


Figure 3.4: The weighted automaton $\mathcal{B}$. In the figure, $a \in \Sigma$ and $b \in \Sigma \setminus \{d\}$.

We are now ready to prove the first result on Attacker-Defender games.

Theorem 3.9. *It is undecidable whether Eve has a winning strategy to reach the target configuration* $[\varepsilon, 0]$ *from a given initial configuration* $[w, 0]$*, where* $w \in \mathrm{FG}(\Gamma_2)$*, in the weighted word game with words over a binary group alphabet* Γ_2*.*

Proof. The proof is based on the reduction of the universality problem for weighted automata on infinite words to the problem of checking for the existence of a winning strategy in the weighted word game. Let $\mathcal{B} = (Q_{\mathcal{B}}, \Sigma, \sigma', q_1, \{q_3\}, \mathbb{Z})$ be the weighted automaton on infinite words from Lemma 3.8.

Let us define the following initial instance of the weighted word game. Adam's moves are just single letters from the alphabet $\Sigma \subset \Gamma$ with weight 0. That is, $A = \{\langle\!\langle a, 0 \rangle\!\rangle \mid a \in \Sigma\}$. Eve has three types of moves. Either she appends the dummy letter $\#$, begins the simulation of $\mathcal{B}$, or continues the simulation. More precisely, Eve's words are over the group alphabet $\Gamma = \Sigma \cup \Sigma^{-1} \cup Q_{\mathcal{B}} \cup Q_B^{-1} \cup \{\#, \overline{\#}\}$ and the move set is

$$E = \{\langle\!\langle \#, 0 \rangle\!\rangle\} \cup \{\langle\!\langle \bar{a}\overline{q_i}, z \rangle\!\rangle \mid \langle q_i, a, q_3, z\rangle \in \sigma'\} \cup \{\langle\!\langle \bar{b}q_j\overline{\#}\bar{a}\overline{q_i}, z \rangle\!\rangle \mid \langle q_i, a, q_j, z\rangle \in \sigma', b \in \Sigma\}.$$

The initial configuration is $[q_3\#, 0]$ and the target configuration is $[q_3\#\overline{q_1}, 0]$. We will later add moves to make the target $[\varepsilon, 0]$ as in our definition of weighted word games.

Then, in the above game, Adam can avoid reaching the configuration $[q_3\#\overline{q_1}, 0]$ if and only if there is an infinite word that is not accepted by the weighted automaton $\mathcal{B}$. Moreover, Eve has to follow the computation path of the infinite word played by Adam in $\mathcal{B}$ in order to reach the target configuration. Note that Eve is simulating $\mathcal{B}$ in reverse, from the final state q_3 to the initial state q_1, following all edges of $\mathcal{B}$ in the opposite direction.

Let us consider how the game progresses in different scenarios. First, let us assume that Adam has played a word p, where $|p| = n$, and Eve decides to simulate $\mathcal{B}$, that is, she wants to show that $pw \in L(\mathcal{B})$ for any $w \in \Sigma^\omega$. Prior to this decision Eve had played only $\#$ and so the current configuration is $[q_3\# \cdot p[1] \cdot \# \cdot p[2] \cdot \ldots \cdot \# \cdot p[n], 0]$.

Let π be a computation path such that $[q, p, z] \in \mathcal{R}(\pi)$ for some $q \in Q_{\mathcal{B}}$ and $z \in \mathbb{Z}$. First, we assume that Eve follows π and show that the target word is reachable if $[q_3, p, 0] \in \mathcal{R}(\pi)$. After which, we show that the target word is not reachable if Eve does not follow π. Let $\langle q_{i_1}, p[1], q_{i_2}, z_1\rangle, \ldots, \langle q_{i_n}, p[n], q_3, z_n\rangle$ be the n elements of π before the computation reaches the final state. After playing moves corresponding to these transitions, the configuration is

$$\begin{bmatrix} q_3\# \\ 0 \end{bmatrix} \left\langle\!\!\left\langle \begin{matrix} p[1] \\ 0 \\ {\scriptstyle A} \end{matrix} \right\rangle\!\!\right\rangle \left\langle\!\!\left\langle \begin{matrix} \# \\ 0 \\ {\scriptstyle E} \end{matrix} \right\rangle\!\!\right\rangle \cdots \left\langle\!\!\left\langle \begin{matrix} p[n] \\ 0 \\ {\scriptstyle A} \end{matrix} \right\rangle\!\!\right\rangle \left\langle\!\!\left\langle \begin{matrix} \overline{p[n]}\overline{q_{i_n}} \\ z_n \\ {\scriptstyle E} \end{matrix} \right\rangle\!\!\right\rangle$$
$$\left\langle\!\!\left\langle \begin{matrix} a_{j_1} \\ 0 \\ {\scriptstyle A} \end{matrix} \right\rangle\!\!\right\rangle \left\langle\!\!\left\langle \begin{matrix} \overline{a_{j_1}} q_{i_n} \overline{\#}\overline{p[n-1]}\overline{q_{i_{n-1}}} \\ z_{n-1} \\ {\scriptstyle E} \end{matrix} \right\rangle\!\!\right\rangle \cdots \left\langle\!\!\left\langle \begin{matrix} a_{j_n} \\ 0 \\ {\scriptstyle A} \end{matrix} \right\rangle\!\!\right\rangle \left\langle\!\!\left\langle \begin{matrix} \overline{a_{j_n}} q_{i_1} \overline{\#}\overline{p[1]}\overline{q_{i_1}} \\ z_1 \\ {\scriptstyle E} \end{matrix} \right\rangle\!\!\right\rangle,$$

where the letter A indicates a move of Adam and the letter E indicates a move of Eve. In the first component, we have the reduced word $q_3\#\overline{q_{i_1}}$ and, in the second component, we have weight $\sum_{i=1}^{n} z_i = z$. Note that any letter a_{j_k}, which is played by Adam after Eve has started the simulation of the automaton, plays no role in the computation as it

is immediately cancelled by Eve. If the configuration $[q_3, p, 0]$ is reachable, then $q_{i_1} = q_1$ and $z = 0$, so the configuration is $[q_3\#\overline{q_{i_1}}, 0]$ and Eve wins. On the other hand, if the configuration $[q_3, p, 0]$ is not reachable, then either $q_{i_1} \neq q_1$ or $z \neq 0$. This means that the current configuration is not the target configuration, therefore Eve has not won (yet).

Next, we show that Eve does not have a winning strategy if she does not follow a computation path of automaton $\mathcal{B}$. We focus on the first component of the game. There are three (not mutually exclusive) cases to consider when Eve does not follow a computation path. Eve can play a move such that, in the reduced word,

- there are at least two inverse letters corresponding to the states of $\mathcal{B}$, or
- there is an inverse letter $\bar{a}$, or
- there is an inverse letter corresponding to a state of $\mathcal{B}$ followed by $\#$.

In the first case, Eve does not have a winning strategy because the available moves of Eve do not decrease the number of inverse letters corresponding to the states. The second case is also straightforward. Eve's moves do not contain letters $a \in \Sigma$, so only Adam can cancel the inverse letter. Therefore, Adam will play $b \neq a$, ensuring that the reduced word $q_3\#\overline{q_1}$ cannot be reached. In the final case, the reduced word is of the form $q_3\#w\overline{q_i}a'\#$. Consider the next turn when Adam plays $a \in \Sigma$ and Eve can only play moves that do not reduce the length of the word. This is clear, as the only moves that cancel the final letter a contain an inverse letter corresponding to the states, after which the reduced word contains two inverse letters corresponding to the states of $\mathcal{B}$ and Eve does not have a winning strategy as shown in the first case.

We have analysed all possible ways Eve can deviate from a faithful simulation of $\mathcal{B}$ and showed that Adam has a winning strategy in them. That is, Eve has a winning strategy if and only if $\mathcal{B}$ is universal.

In order to get a game where the word of a winning configuration is the empty word, rather than $q_3\#\overline{q_1}$, we need to have an extra move for Eve, and to make sure that no false solutions are added. The simple construction of adding words $\bar{a}q_1\overline{\#}\overline{q_3}$ for all $a \in \Sigma$ creates no new solutions as there is no way to reach ε, after $q_3\#$ has been cancelled out and this is the only way to cancel $q_3\#$.

In order to complete the proof, we will require the encoding of Lemma 2.1 between words over an arbitrary group alphabet and a binary group alphabet. The lemma's morphism

gives a way to map words from an arbitrary sized group alphabet into the set of words over a free group alphabet with only two letters. □

We illustrate the construction of the weighted word game from a weighted automaton in the following example.

Example 3.10. Let $\mathcal{A}$ be a weighted automaton depicted in Figure 3.5, where q_0 is the initial state and q_3 is the final state. We construct the corresponding weighted word game. To keep the example clearer, we refrain from doing the final step of the proof. That is, instead of a binary group alphabet, we use a larger group alphabet. Adam has two moves $\langle\!\langle a, 0 \rangle\!\rangle$ and $\langle\!\langle b, 0 \rangle\!\rangle$ and Eve has the following set of 27 moves

$$\left\{ \left\langle\!\left\langle \begin{matrix} \# \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}\overline{q_2} \\ -2 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}\overline{q_2} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}\overline{q_1} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}\overline{q_1} \\ -1 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}\overline{q_0} \\ 2 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}\overline{q_0} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}q_3\overline{\#}\bar{a}\overline{q_2} \\ -2 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_3\overline{\#}\bar{a}\overline{q_2} \\ -2 \end{matrix} \right\rangle\!\right\rangle,$$
$$\left\langle\!\left\langle \begin{matrix} \bar{a}q_3\overline{\#}\bar{b}\overline{q_2} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_3\overline{\#}\bar{b}\overline{q_2} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}q_2\overline{\#}\bar{a}\overline{q_3} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_2\overline{\#}\bar{a}\overline{q_3} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}q_2\overline{\#}\bar{b}\overline{q_3} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_2\overline{\#}\bar{b}\overline{q_3} \\ 0 \end{matrix} \right\rangle\!\right\rangle,$$
$$\left\langle\!\left\langle \begin{matrix} \bar{a}q_3\overline{\#}\bar{a}\overline{q_1} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_3\overline{\#}\bar{a}\overline{q_1} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}q_3\overline{\#}\bar{b}\overline{q_1} \\ -1 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_3\overline{\#}\bar{b}\overline{q_1} \\ -1 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}q_3\overline{\#}\bar{a}\overline{q_0} \\ 2 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_3\overline{\#}\bar{a}\overline{q_0} \\ 2 \end{matrix} \right\rangle\!\right\rangle,$$
$$\left\langle\!\left\langle \begin{matrix} \bar{a}q_3\overline{\#}\bar{b}\overline{q_0} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_3\overline{\#}\bar{b}\overline{q_0} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}q_1\overline{\#}\bar{a}\overline{q_0} \\ 1 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_1\overline{\#}\bar{a}\overline{q_0} \\ 1 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{a}q_0\overline{\#}\overline{q_3} \\ 0 \end{matrix} \right\rangle\!\right\rangle, \left\langle\!\left\langle \begin{matrix} \bar{b}q_0\overline{\#}\overline{q_3} \\ 0 \end{matrix} \right\rangle\!\right\rangle \right\}.$$

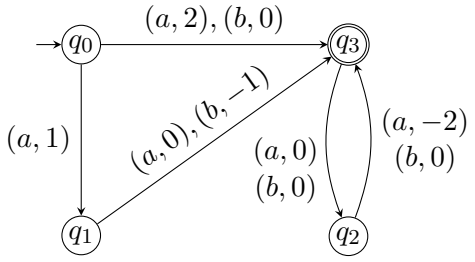


Figure 3.5: Weighted automaton $\mathcal{A}$.

Let us consider a word starting with *abab* and how the weighted word game follows. Starting from the initial configuration $[q_3\#, 0]$, which represents the final state q_3 of $\mathcal{A}$ and weight zero, Adam plays $\langle\!\langle a, 0 \rangle\!\rangle, \langle\!\langle b, 0 \rangle\!\rangle, \langle\!\langle a, 0 \rangle\!\rangle$ and $\langle\!\langle b, 0 \rangle\!\rangle$, while Eve plays $\langle\!\langle \#, 0 \rangle\!\rangle$ thrice until

Eve plays $\langle\!\langle \bar{b}\overline{q_2}, 0 \rangle\!\rangle$ to start the simulation of the automaton:

$$\begin{bmatrix} q_3\# \\ 0 \end{bmatrix} \underset{A}{\left\langle\!\left\langle \begin{matrix} a \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \# \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} b \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \# \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} a \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \# \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} b \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \bar{b}\overline{q_2} \\ 0 \end{matrix} \right\rangle\!\right\rangle}.$$

After this moment of the play, it does not matter which letter Adam plays as Eve can always cancel it. Let $c_1, c_2, c_3, c_4 \in \{a, b\}$ be the letters Adam plays. Now Eve follows the computation path visiting q_1, q_3, q_2 and ending in q_3 in the reverse order:

$$\begin{bmatrix} q_3\# \\ 0 \end{bmatrix} \underset{A}{\left\langle\!\left\langle \begin{matrix} a \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \# \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} b \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \# \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} a \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \# \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} b \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \bar{b}\overline{q_2} \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} c_1 \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \overline{c_1} q_2 \overline{\#} \bar{a} \overline{q_3} \\ 0 \end{matrix} \right\rangle\!\right\rangle} = \begin{bmatrix} q_3\# \cdot a \cdot \# \cdot b \cdot \# \cdot \overline{q_3} \\ 0 \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} q_3\# \cdot a \cdot \# \cdot b \cdot \# \cdot \overline{q_3} \\ 0 \end{bmatrix} \underset{A}{\left\langle\!\left\langle \begin{matrix} c_2 \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \overline{c_2} q_3 \overline{\#} \bar{b} \overline{q_1} \\ -1 \end{matrix} \right\rangle\!\right\rangle} = \begin{bmatrix} q_3\# \cdot a \cdot \# \cdot \overline{q_1} \\ -1 \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} q_3\# \cdot a \cdot \# \cdot \overline{q_1} \\ -1 \end{bmatrix} \underset{A}{\left\langle\!\left\langle \begin{matrix} c_3 \\ 0 \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \overline{c_3} q_1 \overline{\#} \bar{a} \overline{q_0} \\ 1 \end{matrix} \right\rangle\!\right\rangle} = \begin{bmatrix} q_3\# \cdot \overline{q_0} \\ -1+1 \end{bmatrix}.$$

As the weight of this play is 0, Eve wins the game by playing the correct $\langle\!\langle \overline{c_4} q_0 \overline{\#} \overline{q_3}, 0 \rangle\!\rangle$.

Note that $\mathcal{A}$ is not a universal automaton as, for example, aab^ω is not accepted. Thus, Adam has a winning strategy in this game.

3.2.2 Word games on pairs of group words

We now modify the game of the previous subsection by encoding the weight as a separate word over a group alphabet Γ'. This encoding, with additional tricks, allows us to construct a word game where both the initial and final configurations are $[\varepsilon, \varepsilon]$.

This variant of the *word game* consists of Eve and Adam having sets of pairs of words

$$\{\langle\!\langle e_1, e_1' \rangle\!\rangle, \ldots, \langle\!\langle e_r, e_r' \rangle\!\rangle\} \subseteq \mathrm{FG}(\Gamma) \times \mathrm{FG}(\Gamma') \text{ and } \{\langle\!\langle a_1, a_1' \rangle\!\rangle, \ldots, \langle\!\langle a_s, a_s' \rangle\!\rangle\} \subseteq \mathrm{FG}(\Gamma) \times \mathrm{FG}(\Gamma')$$

respectively, where Γ and Γ' are group alphabets. The initial configuration of this game is an element $[w, \varepsilon]$ and the target configuration is the identity element $[\varepsilon, \varepsilon]$. A configuration of a game after t rounds is an element $[w_t, w_t'] = [w, \varepsilon]\langle\!\langle a_{i_1}, a_{i_1}' \rangle\!\rangle\langle\!\langle e_{i_1}, e_{i_1}' \rangle\!\rangle\langle\!\langle a_{i_2}, a_{i_2}' \rangle\!\rangle\langle\!\langle e_{i_2}, e_{i_2}' \rangle\!\rangle \cdot \ldots \cdot \langle\!\langle a_{i_t}, a_{i_t}' \rangle\!\rangle\langle\!\langle e_{i_t}, e_{i_t}' \rangle\!\rangle$, where $\langle\!\langle e_{i_j}, e_{i_j}' \rangle\!\rangle$ and $\langle\!\langle a_{i_j}, a_{i_j}' \rangle\!\rangle$ are elements from above defined sets of Eve and Adam. The decision problem for the word game is to check whether there exists a winning strategy for Eve to reach the identity element $[\varepsilon, \varepsilon]$.

Undecidability of existence of a winning strategy in the word game where elements are generated by $\Gamma \times \{\rho, \bar{\rho}\}$ follows from Theorem 3.9. This follows from the fact that the free group with one generator is isomorphic to the integers $(\mathbb{Z}, +)$. Indeed, consider a move $\langle\!\langle w, z \rangle\!\rangle$ of a player in the weighted word game G. In the word game on pairs of words, the same player has the move $\langle\!\langle w, \rho^z \rangle\!\rangle$. It is easy to see, that the same player has a winning strategy in both games[9]. In the next theorem, we construct a word game where both the initial and target elements are $[\varepsilon, \varepsilon]$. We construct a game where Eve has four copies of moves of G, each encoded over a disjoint group alphabet. We use the idea of the encoding of [14] to ensure that there are four plays of G played in a particular order and $[\varepsilon, \varepsilon]$ is reached in the new game if and only if $[\varepsilon, \varepsilon]$ is reached in all four plays of G.

Theorem 3.11. *It is undecidable whether Eve has a winning strategy to reach* $[\varepsilon, \varepsilon]$ *from* $[\varepsilon, \varepsilon]$ *in the word game where elements are generated by* $\Gamma_2 \times \Gamma_2'$*, where both* Γ_2 *and* Γ_2' *are binary group alphabets, i.e.,* $\Gamma_2 = \{c, d, \bar{c}, \bar{d}\}$ *and* $\Gamma_2' = \{e, f, \bar{e}, \bar{f}\}$.

Proof. Let G be the word game of Theorem 3.9 for which checking whether Eve has a winning strategy is undecidable. As in Example 3.10, we consider the game over a group alphabet Γ before applying the morphism of Lemma 2.1. Let E and A be moves of Eve and Adam of G, where the first component of the move is over a group alphabet Γ and the second component is over the unary group alphabet $\{\rho, \bar{\rho}\}$, and the initial configuration is $[w, \varepsilon]$. Consider the weighted automaton $\mathcal{B}$ of Lemma 3.8 with a single final state, $Q_{\mathcal{B}}$ is the set of states and Σ is its input alphabet.

We want to make sure that the four consecutive plays will be played one after another. For this we introduce eight border letters $\square_1, \square_2, \square_3, \square_4, \diamondsuit_1, \diamondsuit_2, \diamondsuit_3, \diamondsuit_4$ from a fresh group alphabet. Eve's first component consists of words over the group alphabet $\Gamma_1 = \Sigma \cup \Sigma^{-1} \cup \{\#, \overline{\#}\} \bigcup_{k=1}^{4} (Q_{\mathcal{B}_k} \cup Q_{\mathcal{B}_k}^{-1})$ and the second component is over the group alphabet $\{\square_k, \rho_k, \diamondsuit_k, \overline{\square_k}, \overline{\rho_k}, \overline{\diamondsuit_k} \mid k \in \{1, 2, 3, 4\}\}$. We construct Eve's set of moves E' which encode the original moves in E over the four alphabets in the following manner:

- Dummy move: $\langle\!\langle \#, \varepsilon \rangle\!\rangle \in E$ is added to E' as it is;
- Initialisation moves: for each move $\langle\!\langle \bar{a}\overline{q_i}, \rho^z \rangle\!\rangle \in E$, we add moves $\langle\!\langle \bar{a}\overline{q_{i1}}, \square_1 \rho_1^z \diamondsuit_1 \rangle\!\rangle$, $\langle\!\langle \bar{a}\overline{q_{i2}}, \overline{\square_2} \rho_2^z \diamondsuit_2 \rangle\!\rangle$, $\langle\!\langle \bar{a}\overline{q_{i3}}, \overline{\square_3} \rho_3^z \diamondsuit_3 \rangle\!\rangle$, $\langle\!\langle \bar{a}\overline{q_{i4}}, \overline{\square_4} \rho_4^z \diamondsuit_4 \rangle\!\rangle$ to E';

[9] There is an exponential blow-up as the integers are encoded in binary. However, this does not affect the decidability status of the game.

- Simulation moves: for each move $\langle\!\langle \bar{b} q_i \overline{\#} \bar{a} \overline{q_j}, \rho^z \rangle\!\rangle \in E$, where q_j is not the initial state of $\mathcal{B}$, we add moves $\langle\!\langle \bar{b} q_{ik} \overline{\#} \bar{a} \overline{q_{jk}}, \overline{\Diamond_k} \rho_k^z \Diamond_k \rangle\!\rangle$, for each $k \in \{1,2,3,4\}$, to E' ;
- Finishing moves: for each move $\langle\!\langle \bar{b} q_i \overline{\#} \bar{a} \overline{q_j}, \rho^z \rangle\!\rangle \in E$, where q_j is the initial state of $\mathcal{B}$, we add moves $\langle\!\langle \bar{b} q_{ik} \overline{\#} \bar{a} \overline{q_{jk}}, \overline{\Diamond_k} \rho_k^z \Box_{k+1} \rangle\!\rangle$, for each $k \in \{1,2,3\}$, and $\langle\!\langle \bar{b} q_{i4} \overline{\#} \bar{a} \overline{q_{j4}}, \overline{\Diamond_4} \rho_k^z \overline{\Box_1} \rangle\!\rangle$ to E';
- Finally, to finish the game, we add $\langle\!\langle \bar{a} q_{j_4} q_{j_3} q_{j_2} q_{j_1}, \varepsilon \rangle\!\rangle$, where q_j is the initial state of $\mathcal{B}$, to E'.

From the way how the moves are constructed, it follows that the only way to cancel all the border letters (i.e., $\Box_i, \Diamond_i$ and q_{j_i} for $i = 1, \ldots, 4$), is to have four consecutive plays of the game G followed by the move $\langle\!\langle \bar{a} q_{j_4} q_{j_3} q_{j_2} q_{j_1}, \varepsilon \rangle\!\rangle$. Namely, first using the moves containing letters from the alphabet $Q_{\mathcal{B}1}$, then $Q_{\mathcal{B}2}$, followed by $Q_{\mathcal{B}3}$ and finally $Q_{\mathcal{B}4}$, or a cyclic permutation of the order. If the moves are played in a different order, then the border letters will create a non-cancellable pair of elements from two distinct alphabets. It is easy to see that it is impossible to reach $[\varepsilon, \varepsilon]$ afterwards.

Then, if Eve has a winning strategy to reach $[\varepsilon, \varepsilon]$ in the weighted word game G, she also has a winning strategy to reach $[\varepsilon, \varepsilon]$ in the word game on pairs of words. On the other hand, if Adam has a winning strategy in the weighted word game, then no matter how Eve plays each of the four plays, either the first or the second component will remain non-empty (or both). After the whole cycle is played, in the two components, there will be letters over at least four distinct alphabets. Since Eve does not have a winning strategy in G and due to the usage of the border letters, no matter how Eve will play a second cycle, the number of distinct alphabets will not decrease. So, the identity element $[\varepsilon, \varepsilon]$ cannot be generated by a concatenation of four plays of G, unless Eve has a winning strategy in G. A similar idea of encoding generators over four alphabets has been used in [14].

Finally, we encode the words in both components using the monomorphism of Lemma 2.1 to have the game over binary group alphabets Γ_2 and Γ_2'. □

Example 3.12. Consider the weighted word game G of Example 3.10 from which we construct the set E' as in the previous theorem. To illustrate the idea of the encoding, let us consider a prefix of a play

$$\underset{A}{\left\langle\!\left\langle \begin{matrix} a \\ \varepsilon \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \# \\ \varepsilon \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} b \\ \varepsilon \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \bar{b}\overline{q_{11}} \\ \Box_1 \rho_1^{-1} \Diamond_1 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} a \\ \varepsilon \end{matrix} \right\rangle\!\right\rangle} \underset{E}{\left\langle\!\left\langle \begin{matrix} \bar{a} q_{11} \overline{\#} \bar{a} \overline{q_{01}} \\ \overline{\Diamond_1} \rho_1 \Box_2 \end{matrix} \right\rangle\!\right\rangle} \underset{A}{\left\langle\!\left\langle \begin{matrix} a \\ \varepsilon \end{matrix} \right\rangle\!\right\rangle},$$

where moves of Adam are indicated with the letter A and moves of Eve with the letter E. The reduced element of this prefix is $[\overline{q_{01}}a, \square_1\square_2]$. The only moves that cancel $\square_2$ in the second component have letters over $\mathcal{B}_2$ in the first component. That is, a new play of G is simulated using the second alphabet.

3.2.3 Word games over binary group alphabets

The word games of the previous sections can be seen as multidimensional, in the sense that the moves consist of two components. In this subsection, we consider one-dimensional word games, where the words are over binary group alphabets and prove that deciding which player has a winning strategy is in EXPTIME.

The idea of the proof is to construct an alternating pushdown system (PDS), where words played by the players are pushed into the stack letter-by-letter. The cancellation is also possible in the prefix of the word that is pushed, which is checked by the PDS. That is, if a is on top of the stack and a word $\bar{a}u$ is played, then a is popped before the subsequent letters of u are considered in the similar fashion.

Theorem 3.13. *Deciding which player wins in a word game over binary group alphabet is in* EXPTIME.

Proof. Consider a word game over binary group alphabet, where $E \subseteq \mathrm{FG}(\Gamma_2)$ is the move set of Eve, $A \subseteq \mathrm{FG}(\Gamma_2)$ is the move set of Adam and $x_0 \in \mathrm{FG}(\Gamma_2)$ is the initial configuration. We construct an alternating PDS $\mathcal{P}$ for which $L(\mathcal{P}) \neq \emptyset$ if and only if Eve has a winning strategy in the word game.

Let $\mathcal{P} = (Q, \Gamma, \Delta, c_0)$ be an alternating PDS. The state set Q has one state p belonging to $Q_\forall$ and states

$$\{q\} \cup \{e_i[j] \mid e_i \in E \text{ and } j = 1, \ldots, |e_i|\} \cup \{a_i[j] \mid a_i \in A \text{ and } j = 1, \ldots, |a_i| - 1\}$$

belonging to $Q_\exists$. That is, there is a state for each letter in a word of a player. The stack alphabet is $\Gamma = \Gamma_2 \cup \{\bot\}$, where $\bot$ is the special bottom of the stack letter. The set of rules Δ is constructed in such a way, that from q Eve choses which word to play and each word is represented by a path from q to p. In the path, a word is pushed into the stack letter-by-letter taking into account whether the topmost letter in the stack is cancelled.

More formally, the set of rules Δ consists of transitions

$$\begin{aligned}
&\left\{\langle q, x, e_i[1], xe_i[1]\rangle \mid x \neq \overline{e_i[1]}, e_i \in E\right\} \cup \left\{\left\langle q, \overline{e_i[1]}, e_i[1], \varepsilon\right\rangle \mid e_i \in E\right\} \\
&\cup \left\{\langle e_i[j-1], x, e_i[j], xe_i[j]\rangle \mid x \neq \overline{e_i[j]}, e_i \in E, j = 2, \ldots, |e_i|\right\} \\
&\cup \left\{\left\langle e_i[j-1], \overline{e_i[j]}, e_i[j], \varepsilon\right\rangle \mid e_i \in E, j = 2, \ldots, |e_i|\right\} \\
&\cup \{\langle e_i[n], x, p, x\rangle \mid x \in \Gamma_2, e_i \in E, n = |e_i|\} \cup \{\langle e_i[n], \bot, p, \varepsilon\rangle \mid e_i \in E, n = |e_i|\}
\end{aligned}$$

corresponding to moves of Eve in the word game and

$$\begin{aligned}
&\left\{\langle p, x, a_i[1], xa_i[1]\rangle \mid x \neq \overline{a_i[1]}, a_i \in A\right\} \cup \left\{\left\langle p, \overline{a_i[1]}, a_i[1], \varepsilon\right\rangle \mid a_i \in A\right\} \\
&\cup \left\{\langle a_i[j-1], x, a_i[j], xa_i[j]\rangle \mid x \neq \overline{a_i[j]}, a_i \in A, j = 2, \ldots, |a_i| - 1\right\} \\
&\cup \left\{\left\langle a_i[j-1], \overline{a_i[j]}, a_i[j], \varepsilon\right\rangle \mid a_i \in A, j = 2, \ldots, |a_i| - 1\right\} \\
&\cup \left\{\langle a_i[n-1], x, q, xa_i[n]\rangle \mid x \neq \overline{a_i[j]}, A_i \in A, n = |a_i|\right\} \\
&\cup \left\{\left\langle a_i[n-1], \overline{a_i[n]}, q, \varepsilon\right\rangle \mid a_i \in A, n = |a_i|\right\}
\end{aligned}$$

corresponding to moves of Adam in the word game. In the special case where $|a_i| = 1$, rules $\langle p, x, q, xa_i\rangle$, where $x \neq \overline{a_i}$, and $\langle p, \overline{a_i}, q, \varepsilon\rangle$ are added to Δ.

Note that there is a state corresponding to the last letter of Eve's word while for the last letter of Adam's move, the transition is taken to q directly. This is done so that $\bot$ can be removed from a configuration $[e_i[n], \bot]$, where $n = |e_i|$, when transitioning to the accepting configuration $[p, \varepsilon]$.

The initial configuration of $\mathcal{P}$ is $c_0 = [p, \bot x_0]$. It is easy to see that the configuration $[p, \varepsilon]$ can be reached in the constructed alternating PDS if and only if Eve has a winning strategy in the word game. That is, deciding whether Eve has a winning strategy can be done in EXPTIME. □

We illustrate the construction of the alternating PDS in the following example.

Example 3.14. Consider a word game, where $E = \{\bar{b}a, \bar{a}\}$ is the set of moves of Eve, $A = \{a\bar{b}a, \bar{a}\,\bar{a}b\}$ is the set of moves of Adam, and the initial word is $\bar{a}$. Let $\mathcal{P}$ be the alternating PDS constructed as in the proof of the previous theorem. The PDS is depicted in Figure 3.6. A pair of moves of Adam and Eve applied to the initial word, resulting in

$[\bar{a}]\underset{E}{\langle\!\langle a\bar{b}a\rangle\!\rangle}\underset{A}{\langle\!\langle \bar{a}\rangle\!\rangle} = [\bar{b}]$, has the corresponding run in $\mathcal{P}$:

$$[p, \bot\bar{a}] \models [a, \bot] \models [\bar{b}, \bot\bar{b}] \models [q, \bot\bar{b}a] \models [\bar{a}, \bot\bar{b}] \models [p, \bot\bar{b}].$$

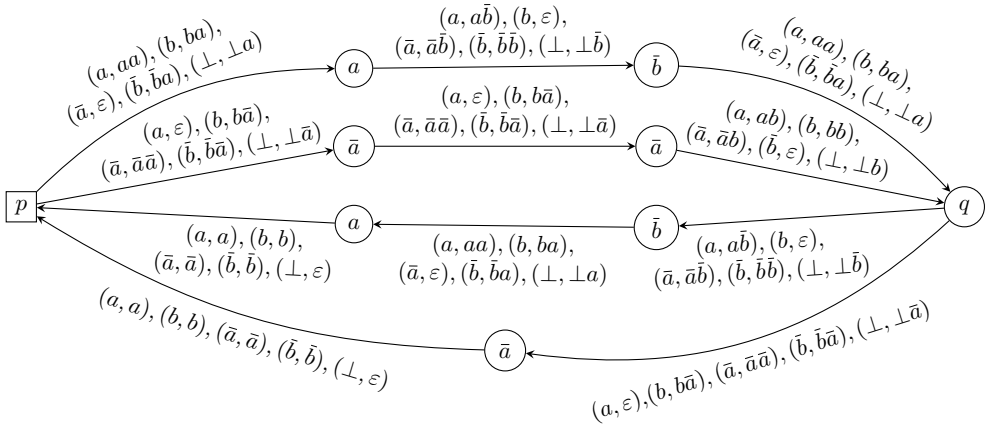


Figure 3.6: An alternating PDS constructed from a word game over a binary alphabet of Example 3.14.

3.2.4 Matrix games on vectors

We extend the domain of the game and a set of rules to the class of linear transformations on integer lattice $\mathbb{Z}^4$. A *matrix game on vectors* (or a *matrix game* for short) consists of two players, Eve and Adam, having sets of linear transformations $\{E_1, \ldots, E_r\} \subseteq \mathbb{Z}^{n\times n}$ and $\{A_1, \ldots, A_s\} \subseteq \mathbb{Z}^{n\times n}$ respectively, an *initial vector* $\mathbf{x}_0 \in \mathbb{Z}^n$ of the game representing the starting configuration, and a *target vector* $\mathbf{y} \in \mathbb{Z}^n$. The *dimension* of the game is the dimension n of the integer lattice.Starting from $\mathbf{x}_0$, players move the current point by applying available linear transformations (by matrix multiplication) from their respective sets in turns. The decision problem of the matrix game is to check whether there exists a winning strategy for Eve to reach the target from the starting configuration (vectors in $\mathbb{Z}^n$) of the game. Note that in our formulation the vectors are horizontal and players multiply them from the right. Recall that $\mathrm{SL}(n, \mathbb{Z}) = \{M \in \mathbb{Z}^{n\times n} \mid \det(M) = 1\}$.

In a game where each player has only one possible move (i.e., when the game is deterministic), the existence of a winning strategy for Eve or Adam can be trivially reduced to the *orbit problem* by combining the two matrices into one. The orbit problem is

decidable in polynomial time for any matrix size over integers, rationals and even algebraic numbers [93].

If Adam has a single move, the problem of checking whether Eve has a winning strategy corresponds to the standard vector reachability problem which has been extensively studied in the literature [12, 13, 22, 73, 81, 112, 132]. Indeed, let $E = \{E_1, \dots, E_k\}$ be a matrix set for which the vector reachability problem is undecidable (for vectors $\mathbf{x}, \mathbf{y}$) and $A = \{A_1\}$ ($A_1 \in \mathrm{SL}(n, \mathbb{Z})$). Then let us consider the matrix game with Eve's set $\{A_1^{-1}E_1, \dots, A_1^{-1}E_k\}$, Adam's set A, the initial vector is $\mathbf{x}$, and the target vector is $\mathbf{y}$. In this game, Eve has a winning strategy if and only if $\mathbf{y}$ is reachable from $\mathbf{x}$ in the vector reachability problem. Thus, a game where Eve has six matrices in $\mathbb{Z}^{3\times 3}$, and Adam has a single matrix from $\mathrm{SL}(3, \mathbb{Z})$ is undecidable, following the undecidability of the vector reachability problem for six integer matrices in dimension three [81].

In the symmetric case when Eve has a single matrix and Adam has m matrices, we can reduce the problem of checking for the existence of a winning strategy to a different reachability problem for matrix semigroups with a stronger reachability objective. That is, Eve's set is $E = \{E_1\}$, Adam's set is $\{A_1, \dots, A_m\}$, the initial vector $\mathbf{x}$, and the target vector $\mathbf{y}$. Again, we can combine sets E and A into one generating set $A' = \{A_1E_1, \dots, A_mE_1\}$ of a semigroup S. However, the question whether Eve has a winning strategy would require for us to check that, on any infinite trajectory of reachable points starting from the initial point $\mathbf{x}$ and transformed by elements of S, the point $\mathbf{y}$ is eventually reachable. Let us now formally define this problem.

Problem 3.15. *Let S be a matrix semigroup generated by $A = \{A_1, \dots, A_m\}$ and $\mathbf{x}, \mathbf{y} \in \mathbb{Z}^n$. In the* eventual reachability problem*, we are asked whether for every element $M = A_{i_1} \cdots A_{i_k}$ either there exists $j \leq k$ such that $\mathbf{x}A_{i_1} \dots A_{i_j} = \mathbf{y}$ or there exists $N \in S$ such that $\mathbf{x}MN = \mathbf{y}$. In other words, $\mathbf{y}$ appears in every trajectory starting from $\mathbf{x}$.*

To the author's best knowledge, this problem has not been studied previously. The problem is illustrated in Figure 3.7. The solution of this problem gives the answer to whether a winning strategy exists in a matrix game where Eve has one move and Adam has several moves.

Let us consider the case where both Eve and Adam have at least two moves. Let us assume that $A = \{A_1, A_2\} \subseteq \mathrm{SL}(n, \mathbb{Z})$ and $E = \{E_1, E_2\}$. Consider a game where Adam's

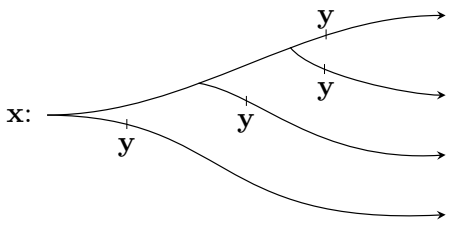


Figure 3.7: An illustration of traces in the eventual reachability problem.

set is A and Eve's set is

$$E' = \{E_1A_1^{-1}, E_1A_2^{-1}, E_2A_1^{-1}, E_2A_2^{-1}\}.$$

The previous reasoning of reducing the problem to the standard vector reachability for matrix semigroups is not directly applicable. The reachability of $\mathbf{y}$ from $\mathbf{x}$ using matrices from A implies that there exists a winning strategy in the matrix game. Eve's strategy is to follow the solution of the reachability problem with the moves that cancel the matrix played by Adam (i.e., if Adam played A_i, then Eve plays $E_jA_i^{-1}$). On the other hand, the existence of a winning strategy does not imply that $\mathbf{y}$ is reachable from $\mathbf{x}$. Indeed, unlike previously, Eve might follow A_1 with $A_2^{-1}E_1$ and can still reach the target position. If we try to consider the reachability using matrices from E', containing all the possible moves of Adam followed by all the possible moves of Eve, then from the fact that $\mathbf{y}$ is reachable from $\mathbf{x}$, it would not follow that Eve has a winning strategy. Indeed, this kind of trick eliminates the role of Adam and says very little in relation to the game.

Next, we prove the main theorem regarding the matrix game where both Eve and Adam have at least two moves.

Theorem 3.16. *Given two finite sets of matrices* $\{E_1, E_2, \ldots, E_r\} \subseteq \mathbb{Z}^{n\times n}$ *for* Eve *and* $\{A_1, A_2, \ldots, A_s\} \subseteq \mathbb{Z}^{n\times n}$ *for* Adam*, where* $r, s \geq 2$*, an initial starting vector* $\mathbf{x}_0 \in \mathbb{Z}^n$ *and a target vector* $\mathbf{y} \in \mathbb{Z}^n$*, it is undecidable whether Eve has a winning strategy in the matrix game. Furthermore, the claim holds even when the matrices are from* $\mathrm{SL}(4, \mathbb{Z})$*.*

Proof. Let $\Gamma_2 = \{c, d, \bar{c}, \bar{d}\}$ be a binary group alphabet and define $f : \mathrm{FG}(\Gamma_2) \to \mathrm{SL}(2, \mathbb{Z})$ by: $f(c) = \left(\begin{smallmatrix} 1 & 2 \\ 0 & 1 \end{smallmatrix}\right), f(\bar{c}) = \left(\begin{smallmatrix} 1 & -2 \\ 0 & 1 \end{smallmatrix}\right), f(d) = \left(\begin{smallmatrix} 1 & 0 \\ 2 & 1 \end{smallmatrix}\right), f(\bar{d}) = \left(\begin{smallmatrix} 1 & 0 \\ -2 & 1 \end{smallmatrix}\right)$.

Then mapping f is a monomorphism [15] and $f(\varepsilon)$ corresponds to the identity matrix in $\mathbb{Z}^{2\times 2}$. Let α be a function defined in Lemma 2.1, then by the following straightforward

matrix multiplication we have:

$$f(\alpha(z_j)) = f(c^j d \bar{c}^j) = \begin{pmatrix} 1+4j & -8j^2 \\ 2 & 1-4j \end{pmatrix}.$$

Let us show that if $(1,0)M = (1,0)$, where M is an image of a word over binary group alphabet under f, that is, $M \in \{f(\alpha(w)) \mid w \in \mathrm{FG}(\Gamma_2)\}$, then M is the identity matrix. The similar reasoning was used in [15] to prove that the only matrix with zero in the upper corner is the identity matrix. Let $M = \left(\begin{smallmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{smallmatrix}\right)$, now $(1,0)M = (m_{11}, m_{12})$, which implies that $m_{11} = 1$ and $m_{12} = 0$. By the previous observation, $m_{22} = 1$. The final letter of $\alpha(w)$ is $\bar{c}$, which is $\left(\begin{smallmatrix} 1 & -2 \\ 0 & 1 \end{smallmatrix}\right)$ under f. Let $Y = f(\alpha(w))f(\bar{c})^{-1} = \left(\begin{smallmatrix} x & y \\ z & v \end{smallmatrix}\right)$. Now $f(\alpha(w)) = \left(\begin{smallmatrix} x & y \\ z & v \end{smallmatrix}\right)\left(\begin{smallmatrix} 1 & -2 \\ 0 & 1 \end{smallmatrix}\right) = \left(\begin{smallmatrix} x & y-2x \\ z & v-2z \end{smallmatrix}\right)$. Since $x = 1$, $y - 2x = 0$ and $x - 2x = 1$, we see that $f(\alpha(w)) = \left(\begin{smallmatrix} 1 & 0 \\ z & 1 \end{smallmatrix}\right) = f(d)^{z/2}$ but by the definition of the encoding this is possible only when $z = 0$. This implies that $f(\alpha(w))$ is the identity matrix.

Let us encode the word game into the matrix game. Recall that by Theorem 3.11, it is undecidable whether Eve has a winning strategy to reach $[\varepsilon, \varepsilon]$ from $[\varepsilon, \varepsilon]$ in a word game. We construct 4×4 matrices with words of the first component encoded by f in the upper left corner and words from the second components encoded by f in the lower right corner. The direct application of the above function to the elements of the word game gives us a set of matrices for Eve and a set of matrices for Adam from $\mathrm{SL}(4,\mathbb{Z})$. By previous considerations for vector $x_0 = (1,0,1,0)$, the equation $x_0 = x_0 \cdot M$, where $M \in \mathrm{SL}(4,\mathbb{Z})$ has only one matrix M satisfying the above statement, the identity matrix in $\mathbb{Z}^{4\times 4}$. Therefore, for every matrix game, where the initial vector x_0 is $(1,0,1,0)$, the question about the winning strategy of reaching x_0 is equivalent to the question of reaching the identity matrix in the product with alternation in applications of Adam's and Eve's linear transformations, which in its turn corresponds to reaching the identity element in the word game. □

Example 3.17. Consider a matrix game, where Adam has matrices $\left(\begin{smallmatrix} 3 & 5 \\ 1 & 2 \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} -1 & -1 \\ 0 & -1 \end{smallmatrix}\right)$ and Eve has matrices $\left(\begin{smallmatrix} 1 & -2 \\ -2 & 3 \end{smallmatrix}\right)$, $\left(\begin{smallmatrix} 1 & -2 \\ 3 & -7 \end{smallmatrix}\right)$, $\left(\begin{smallmatrix} 7 & -17 \\ -5 & 12 \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} 1 & -2 \\ 3 & -7 \end{smallmatrix}\right)$.

Consider a play from $\mathbf{x}_0 = (1,0)$, where Adam plays $M_1 = \left(\begin{smallmatrix} 3 & 5 \\ 1 & 2 \end{smallmatrix}\right)$ followed by $M_3 = \left(\begin{smallmatrix} -1 & -1 \\ 0 & -1 \end{smallmatrix}\right)$, and Eve plays $M_2 = \left(\begin{smallmatrix} 1 & -2 \\ -2 & 3 \end{smallmatrix}\right)$ followed by $M_4 = \left(\begin{smallmatrix} 1 & -2 \\ 3 & -7 \end{smallmatrix}\right)$. That is, the following product

$$(1,0)\begin{pmatrix} 3 & 5 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 1 & -2 \\ -2 & 3 \end{pmatrix}\begin{pmatrix} -1 & -1 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} 1 & -2 \\ 3 & -7 \end{pmatrix} = (1,0).$$

From this computation, we see that the play is winning for Eve. With similar computations, we can see that in fact Eve has a winning strategy in this game. The play $\mathbf{x}_0 M_1 M_2 M_3 M_4$ is depicted in Figure 3.8.

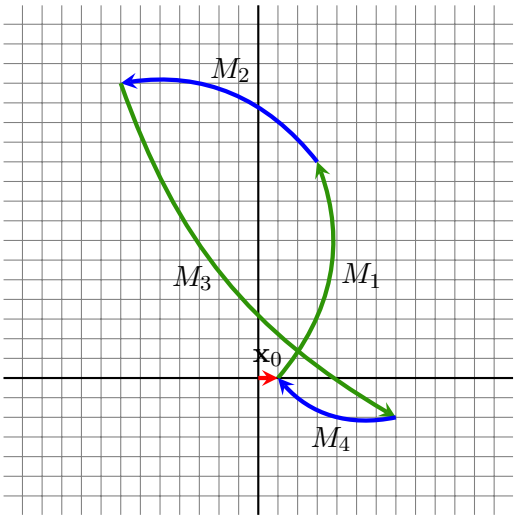


Figure 3.8: The play $\mathbf{x}_0 M_1 M_2 M_3 M_4$ of a matrix game. Green arrows represent transformations by Eve and blue by Adam.

Consider the matrices constructed in the proof of the previous theorem. The moves of the players are block diagonal matrices, i.e., of the form $\left(\begin{smallmatrix} A & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & B \end{smallmatrix}\right)$, where $A, B \in \mathrm{SL}(2, \mathbb{Z})$ and $\boldsymbol{O}_2$ is the zero matrix. Thus, we have the following corollary.

Corollary 3.18. *It is undecidable which player has a winning strategy in a four-dimensional matrix game on vectors, where players' moves are nontrivial block diagonal matrices*[10].

The result of Theorem 3.16 can be improved. By encoding moves of a $\mathbb{Z}^2$-VAS game into $\mathbb{Z}^{3\times 3}$ matrices, we can show that the matrix game is undecidable even in dimension three.

Theorem 3.19. *Given two finite sets of matrices* $\{E_1, E_2, \ldots, E_r\} \subseteq \mathbb{Z}^{3\times 3}$ *for* Eve *and* $\{A_1, A_2, \ldots, A_s\} \subseteq \mathbb{Z}^{3\times 3}$ *for* Adam*, an initial starting vector* $\mathbf{x}_0 \in \mathbb{Z}^3$ *and a target vector* $\mathbf{y} \in \mathbb{Z}^3$*, it is undecidable whether Eve has a winning strategy in the matrix game.*

[10] We say that an n-dimensional block diagonal matrix is trivial if it consists of one block matrix of size n.

Proof. Let $(A, E, (x, y))$ be a $\mathbb{Z}^2$-VAS game. We construct a matrix game, where the initial vector is $\mathbf{x}_0 = (x, 1, y)$ and the moves of the players are mapped into matrices by $f : \mathbb{Z}^2 \to \mathbb{Z}^{3\times 3}$:

$$(a, b) \mapsto \begin{pmatrix} 1 & 0 & 0 \\ a & 1 & b \\ 0 & 0 & 1 \end{pmatrix}.$$

It is easy to see that $(a, b)+(c, d) = (a+c, b+d)$ is correctly simulated by $f((a, b))+f((c, d))$. The target $\mathbf{y}$ of the matrix game is $(0, 1, 0)$. Clearly, Eve has a winning strategy to reach $\mathbf{y}$ from $\mathbf{x}_0$ if and only if she has a winning strategy in $\mathbb{Z}^2$-VAS game, which we will prove to be undecidable in Corollary 5.13. □

The matrices from the previous proof are not block diagonal, so Corollary 3.18 remains the state-of-art undecidabilty result for matrix games with block diagonal moves.

Let us then consider a two-dimensional game. The block diagonal matrices in $\mathbb{Z}^{2\times 2}$ are exactly diagonal matrices, i.e., of the form $\left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right)$, where $a, b \in \mathbb{Z}$. The linear transformation by a diagonal matrix is simple: $\left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right)\left(\begin{smallmatrix} x \\ y \end{smallmatrix}\right) = \left(\begin{smallmatrix} ax \\ by \end{smallmatrix}\right)$. From this we see that two-dimensional diagonal block matrix game is finite as after each turn the absolute values of components of the configuration vector increase.

Theorem 3.20. *It is decidable in polynomial time which player has a winning strategy in two-dimensional diagonal block matrix games.*

Proof. Let $(A, E, (x_0, y_0), (x_f, y_f))$ be a two-dimensional diagonal matrix game, where $A, E \subseteq \{\left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right) \mid a, b \in \mathbb{Z}\}$, $(x_0, y_0) \in \mathbb{Z}^2$ is the initial vector and $(x_f, y_f) \in \mathbb{Z}^2$ is the target vector. Denote by $a_1 = \min\{|a| \mid \left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right) \in A\}$ and $a_2 = \min\{|b| \mid \left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right) \in A\}$, and analogously, $e_1 = \min\{|a| \mid \left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right) \in E\}$ and $e_2 = \min\{|b| \mid \left(\begin{smallmatrix} a & 0 \\ 0 & b \end{smallmatrix}\right) \in E\}$. Let q be the smallest integer such that either $|x_f| \leq (e_1 a_1)^q |x_0|$ or $|y_f| \leq (e_2 a_2)^q |y_0|$. That is, q is the maximal number of turns the game can proceed. Now we can construct a finite game graph with $q|A||E|$ nodes and determine the winner using the standard attractor construction in polynomial time. □

3.2.5 Braid games

The domain of computational games is not limited by considering games on integers, words or matrices. There is also recent interest about the complexity and termination of the games

on braids [26, 34] that are defined with specific rules of adding and removing crossings. In this subsection, we consider the Attacker-Defender games on topological objects, braids in B_n. The moves of the game are compositions of braids in B_3 (a class of braids with only three strands) and B_5 (a class of braids with only five strands) [133]. Braids are classical topological objects that attracted a lot of attention due to their connections to topological knots and links, as well as their applications to polymer chemistry, molecular biology, cryptography, quantum computations and robotics [51, 59, 66, 72, 129]. In this section, we consider very simple games on braids with only three or five strands (i.e., B_3 or B_5) where the braid is modified by a composition with a finite set of braids. We show that it is undecidable to check for the existence of a winning strategy in such games, while the reachability with a single-player (i.e., with nondeterministic composition from a single set) was shown to be decidable for B_3 and undecidable for B_5 in [133].

Definition 3.21. The n-strand braid group B_n is the group given by the presentation with $n-1$ generators $\sigma_1, \ldots, \sigma_{n-1}$ and the following relations $\sigma_i\sigma_j = \sigma_j\sigma_i$, for $|i-j| \geq 2$ and $\sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1}$ for $1 \leq i \leq n-2$. These relations are called Artin's relations.

Elements of the braid group B_n will be represented by words in the alphabet

$$\{\sigma_1, \ldots, \sigma_{n-1}, \sigma_1^{-1}, \ldots, \sigma_{n-1}^{-1}\}$$

and we refer to them as braid words[11].

A composition of two braids with four strands is illustrated in Figure 3.9.

The *fundamental braid* of B_n is

$$\Delta_n = (\sigma_{n-1}\sigma_{n-2}\cdots\sigma_1)(\sigma_{n-1}\sigma_{n-2}\cdots\sigma_2)\cdots\sigma_{n-1}.$$

Geometrically, the fundamental braid is obtained by lifting the bottom ends of the identity braid and flipping (right side over left) while keeping the ends of the strings in a line.

The braid game can be defined in a way, where the sets of braid words $\{e_1, \ldots, e_r\}$ and $\{a_1, \ldots, a_s\}$, for Eve and Adam respectively, will correspond to braids in B_n. The initial braid of the game is given and each following configuration of the game is changed by Eve or Adam by composing braids from their corresponding sets. Given two geometric braids, we can compose them, i.e., put one after the other making the endpoints of the

[11] Whenever a crossing of strands i and $i+1$ is encountered, either σ_i or ${\sigma_i}^{-1}$ is written down, depending on whether the strand i moves under or over the strand $i+1$.

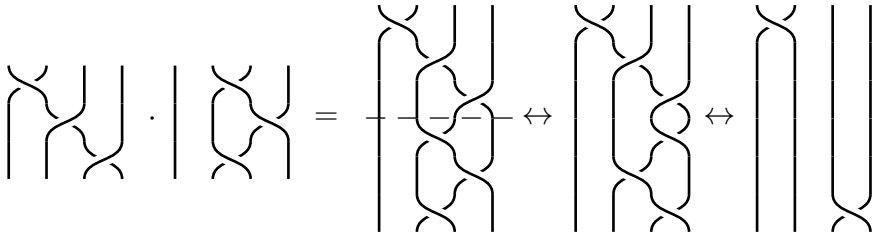


Figure 3.9: An example of a composition of braids in B_4.

first one coincide with the starting points of the second one. There is a neutral element for the composition: it is the trivial braid, also called the identity braid, i.e., the class of the geometric braids where all the strings are straight. Two geometric braids are isotopic if there is a continuous deformation of the ambient space that deforms one into the other, by a deformation that keeps every point in the two bordering planes fixed.

Finally, the goal of Eve is to unbraid, i.e., to reach a configuration of the game that is isotopic to the trivial braid (empty word) and Adam tries to keep Eve from reaching it. Two braids are isotopic if their braid words can be translated one into each other via the relations from Definition 3.21 plus the relations $\sigma_i\sigma_i^{-1} = \sigma_i^{-1}\sigma_i = 1$, where 1 is the identity (trivial braid).

Theorem 3.22. *The braid game is undecidable for braids from B_3 starting from a nontrivial braid and for braids from B_5 starting from the trivial braid.*

Proof. We encode the undecidable weighted word game of Theorem 3.9 into a braid game with three strands and the undecidable word game of Theorem 3.11 into a braid game with five strands and show that the respective braid games are undecidable as well.

Let $\Gamma_2 = \{c, d, \bar{c}, \bar{d}\}$ be a binary group alphabet and define $f : \mathrm{FG}(\Gamma_2) \to B_3$ by: $f(c) = {\sigma_1}^4, f(\bar{c}) = {\sigma_1}^{-4}, f(d) = {\sigma_2}^4, f(\bar{d}) = {\sigma_2}^{-4}$. Then mapping f is a monomorphism [18]. Let α be the mapping from Lemma 2.1. Then:

$$f(\alpha(z_j)) = f(c^j d \bar{c}^j) = {\sigma_1}^{4j}{\sigma_2}^4{\sigma_1}^{-4j}$$

and the length of a braid word from B_3 corresponding to a letter $z_j \in \Gamma$ is $8j + 4$. The above morphisms give a way to map words from an arbitrary sized group alphabet into the set of braid words in B_3.

Now we again can use the weighted word game as any word over a binary group alphabet can be uniquely mapped into a braid, where the empty word will correspond to a braid which is isotopic to the trivial braid and the concatenation of words over group alphabet corresponds to the composition of braids in B_3. The weight $x \in \mathbb{Z}$ is mapped into the braid word Δ_3^{2x}, where $\Delta_3^2 = (\sigma_1\sigma_2\sigma_1)^2$ is a central element of B_3.

Subgroups $\langle \sigma_1^4, \sigma_2^4 \rangle, \langle \sigma_4^2, d \rangle$ of the group B_5 are free and B_5 contains the direct product $\langle \sigma_1^4, \sigma_2^4 \rangle \times \langle \sigma_4^2, d \rangle$ of two free groups of rank two as a subgroup, where $d = \sigma_4\sigma_3\sigma_2\sigma_1^2\sigma_2\sigma_3\sigma_4$ [18]. Now we can uniquely encode pairs of words of the word game into B_5. Using the word game, where the initial position is $[\varepsilon, \varepsilon]$, we can construct a braid game from B_5 starting from the trivial braid.

It is easy to see that Eve has a winning strategy in a braid game on B_3 and B_5 if and only if she has a winning strategy in a weighted word game and a word game on pairs of group words, respectively. □

Example 3.23. Consider a braid game on B_3, where Adam has braid words $\sigma_1\sigma_2^{-1}$ and $\sigma_1^{-1}\sigma_2^{-1}\sigma_1^{-1}$ and Eve has braid words $\sigma_2\sigma_1\sigma_2$ and $\sigma_2\sigma_1$. Starting from $\sigma_1^{-1}\sigma_1^{-1}$, we have the following play:

$$\begin{aligned}\left[\sigma_1^{-1}\sigma_1^{-1}\right] \underset{A}{\langle\!\langle \sigma_1\sigma_2^{-1} \rangle\!\rangle} \underset{E}{\langle\!\langle \sigma_2\sigma_1\sigma_2 \rangle\!\rangle} \underset{A}{\langle\!\langle \sigma_1^{-1}\sigma_2^{-1}\sigma_1^{-1} \rangle\!\rangle} \underset{E}{\langle\!\langle \sigma_2\sigma_1 \rangle\!\rangle} &= \left[\sigma_2\sigma_1^{-1}\sigma_2^{-1}\sigma_1^{-1}\sigma_2\sigma_1\right] \\ &= \left[\sigma_2\sigma_2^{-1}\sigma_1^{-1}\sigma_2^{-1}\sigma_2\sigma_1\right] = [1]\,,\end{aligned}$$

where the second equality follows from the relation $\sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma_2$ of Definition 3.21. This play is depicted in Figure 3.10.

From the definition of braids, braids with two strands represent integers with a braid word σ_1^z corresponding to an integer $z \in \mathbb{Z}$. From the decidability of robot games in dimension one [9], it follows that the braid game on B_2 is also decidable. The braid game on B_3 is the first nontrivial case that is undecidable. In B_5, the game starting from the trivial braid was shown to be undecidable.

3.3 Concluding remarks and open problems

The results of the chapter are twofold. We have proven a new language-theoretic result for weighted automata on infinite words. We constructed an automaton that, for a given instance of the ωPCP, accepts all the infinite words that are not the solutions of the

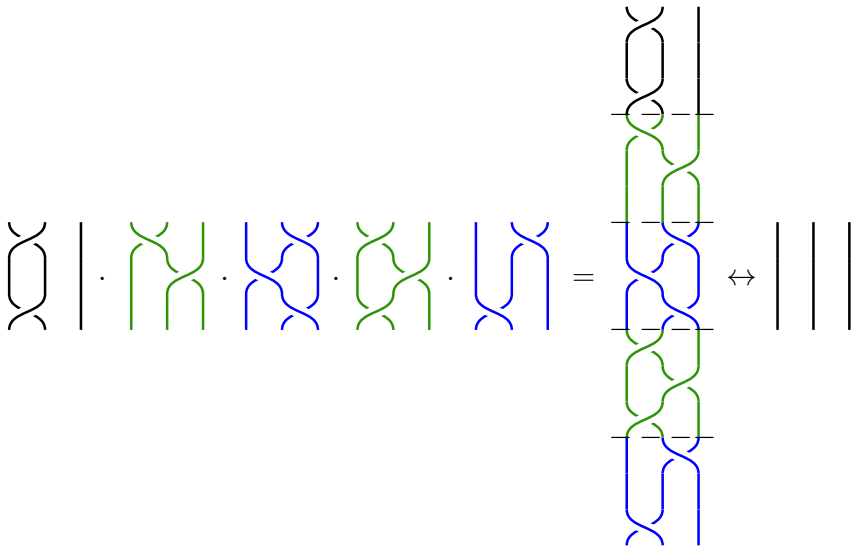


Figure 3.10: An example of a braid game. Green braids represent braids played by Eve and blue braids played by Adam.

instance of the ωPCP. In other words, the non-universality of the automaton corresponds to the instance of the ωPCP having a solution. Secondly, we have shown how to encode the automaton into the framework of Attacker-Defender games, from which we obtained undecidability results for checking for the existence of a winning strategy in word games, matrix games on vectors and braid games.

For weighted automata on infinite words, the status of the universality problem remains open for automata with two states. For the matrix game on vectors, it is unknown whether deciding the winner is decidable for dimensions two and for dimension three if the moves are block diagonal matrices. The status of the braid games on B_3 and B_4 starting from the trivial braid are open. However, the direct encoding of the word game is not applicable due to the fact that there is no faithful representation of the direct product of two free groups of rank two into B_4 [4].

Chapter 4

One-dimensional $\mathbb{Z}$-VASS games

In this chapter, we consider Attacker-Defender games, called $\mathbb{Z}$-VASS games, with simpler moves than in the previous chapter. In Chapter 3, we proved that for games, where the moves are matrices or pairs of words, the problem of checking which player has a winning strategy is undecidable. In the games of this chapter, the players' moves consist of integers with which they modify the counter. As mentioned in subsection 3.2.2, integers can be considered as words over unary alphabet. While the moves are simpler, we add internal state structure for the players. As Attacker-Defender games are turn-based, the states of players are disjoint.

That is, in this chapter, we consider one-dimensional $\mathbb{Z}$-VASS games. Our main result is to prove that $\mathbb{Z}$-VASS games in dimension one are EXPSPACE-complete by presenting a mutual reduction between $\mathbb{Z}$-VASS games and counter reachability games. Note that this is not obvious as the games have essential differences. In a counter reachability game, since the game is played on a graph, a choice of a player, say Eve, affects from which state Adam moves next. In fact, it is not guaranteed that Adam will move at all, as it is possible for Eve to move only between her states. On the other hand, in $\mathbb{Z}$-VASS games, the next state of a player is determined only by his or her previous move.

In order to show EXPSPACE-hardness, we construct a one-dimensional $\mathbb{Z}$-VASS game that can simulate a given one-dimensional counter reachability game such that Eve has a winning strategy in the $\mathbb{Z}$-VASS if and only if she has a winning strategy in the counter reachability game. The idea is for Eve to simulate the whole graph of the counter reachability game and for Adam to verify that Eve is simulating it correctly. In the constructed $\mathbb{Z}$-VASS game Eve has $n + 1$ states, where the counter reachability game has n vertices, and Adam

has only one state. Then, to show completeness, we transform a ℤ-VASS game into a counter reachability game such that Eve has a winning strategy in the counter reachability game if and only if she has a winning strategy in the ℤ-VASS game. The construction is relatively simple and involves storing information on the state of a player into the states of the opponent.

Seeing how adding states for Eve increases the complexity of deciding the winner from EXPTIME of ℤ-VAS games to EXPSPACE of ℤ-VASS games, we consider a state structure of Adam that does not increase the complexity of the game. We show that deciding the winner in one-dimensional ℤ-VASS game where Eve is stateless and Adam's states are *flat*, is in EXPTIME. Flat automata have been studied in various contexts [52–54, 108, 111] and have been shown to be a fruitful tool in verification of counter automata. Flat automata is a subclass of automata where the automaton does not have nested loops. This particular structure allows us to break a ℤ-VASS game into several stateless games that can be solved in EXPTIME. The main challenge is in connecting these separate games. As Adam's underlying state structure is flat, there are only finitely many transitions from one game to another. This fact together with the particular structure of winning sets constructed by the algorithm for a stateless game of [9] provide us with necessary tools to decide the winner in EXPTIME.

In Table 4.1 is a summary of complexity results on one-dimensional ℤ-VASS games according to the state structure of each player. Most of the variants are EXPSPACE-complete, only ℤ-VAS games (i.e., both players are stateless) and ℤ-VASS games where Adam has flat states are EXPTIME-complete. The only variant without a tight complexity is ℤ-VASS games where Eve is stateless and Adam has arbitrary state structure.

Finally, we consider one-dimensional VAS games and show that restricting configurations to positive half-line does not change the complexity and the games remain EXPTIME-complete. The results are summarised in Table 4.2.

4.1 ℤ-VASS games in dimension one

In this section, we consider ℤ-VASS games in dimension one. First, we recall some known results.

Theorem 4.1 (Hunter [88])**.** *Deciding which player wins in a one-dimensional counter reachability game is* EXPSPACE*-complete.*

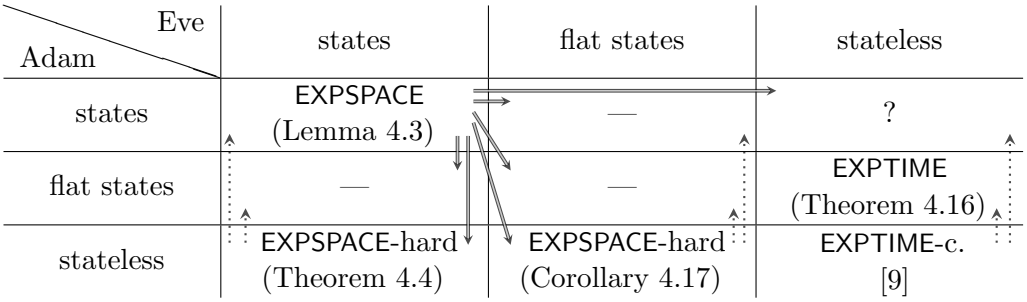


Table 4.1: Complexity of checking for the existence of a winning strategy for Eve in different variants of one-dimensional $\mathbb{Z}$-VASS games. The propagation of upper bounds is depicted with double arrows and of lower bounds with dotted arrows.

Game	VAS	$\mathbb{Z}$-VAS	$\mathbb{Z}$-VASS
Complexity	EXPTIME-complete (Theorem 4.19)	EXPTIME-complete [9]	EXPSPACE-complete (Theorem 4.4)

Table 4.2: Complexity of checking for the existence of a winning strategy for Eve in different one-dimensional games.

Theorem 4.2 (Arul, Reichert [9])**.** *Deciding which player wins in a one-dimensional robot game is* EXPTIME*-complete.*

In our terminology, robot games are $\mathbb{Z}^d$-VAS games, and since they are a special case of $\mathbb{Z}^d$-VASS games, we can inherit the lower bound. That is, the $\mathbb{Z}$-VASS are EXPTIME-hard. On the other hand, it is easy to construct a counter reachability game out of a $\mathbb{Z}$-VASS game by storing information on the state of Eve in the $\mathbb{Z}$-VASS game in the states of Adam and vice versa. That is, $\mathbb{Z}$-VASS games are in EXPSPACE.

Lemma 4.3. *Deciding which player wins in a one-dimensional* $\mathbb{Z}$*-*VASS *game is in* EXPSPACE*.*

Proof. Let (A, E, c_0) be a $\mathbb{Z}$-VASS game. We construct a counter reachability game $((V, F), c_0')$ where Eve has a winning strategy if and only if Eve has a winning strategy in (A, E, c_0). Eve's states are $V_E = \{s_t \mid s \in Q_E, t \in Q_A\}$ and Adam's states are $V_A = \{t_s \mid t \in Q_A, s \in Q_E\}$. The edges of the graph are

$$F = \{(s_t, z, t_{s'}) \mid \langle\!\langle s, z, s' \rangle\!\rangle \in E\} \cup \{(t_s, z, s_{t'}) \mid \langle\!\langle t, z, t' \rangle\!\rangle \in A\}.$$

The construction is depicted in Figure 4.1. Let $c_0 = [s_0, t_0, z_0]$, then the initial configuration of the counter reachability game is $c'_0 = [s_{0t_0}, z_0]$. It is clear that Eve has a winning strategy in (A, E, c_0) if and only if Eve has a winning strategy in $((V, F), c'_0)$. As deciding the winner in the one-dimensional counter reachability game is EXPSPACE-complete, also deciding the winner in the $\mathbb{Z}$-VASS game is in EXPSPACE. □

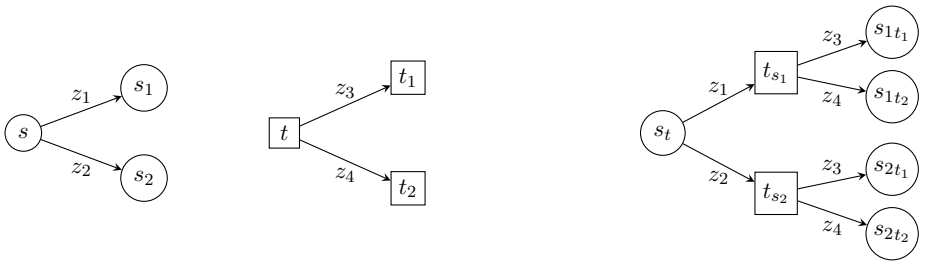


Figure 4.1: Moves in a $\mathbb{Z}$-VASS game (left) and the corresponding part of the graph of the counter reachability game (right).

We provide the matching tight lower bound, showing that one-dimensional $\mathbb{Z}$-VASS games are EXPSPACE-complete. That is, we show that the $\mathbb{Z}$-VASS games are EXPSPACE-hard. To prove this, we show how, for any counter reachability game, to construct a $\mathbb{Z}^d$-VASS game such that the same player wins in both games. The idea is for Eve to have the whole graph, including Adam's states, as her states and Adam to have a single state. Adam has three moves, two to tell Eve which edge to pick if the state was initially Adam's, and one to do nothing if that's not the case.

Theorem 4.4. *One-dimensional* $\mathbb{Z}$-VASS *games are* EXPSPACE*-complete.*

First, we consider a simple modification to a counter reachability game. We can assume that in every Adam's state there are at most two outgoing edges. Indeed, let t be Adam's vertex with k outgoing edges, we replace it by a chain of vertices $t_1, \ldots, t_k$ such that ith edge (t, z, r) is (t_i, z, r). Finally, we connect the vertices with edges $(t_i, 0, t_{i+1})$ for $i \in \{1, \ldots, k-1\}$ and $(t, 0, t_1)$. This construction is depicted in Figure 4.2.

Next, we show the gadgets for different moves in the one-dimensional counter reachability games. At this state, for simplicity, we assume that both players will play in good faith and will simulate the counter reachability game correctly. Later on, we'll construct an additional gadget for Eve and show that if one of the player cheats, then the other can catch the cheating player and has a winning strategy.

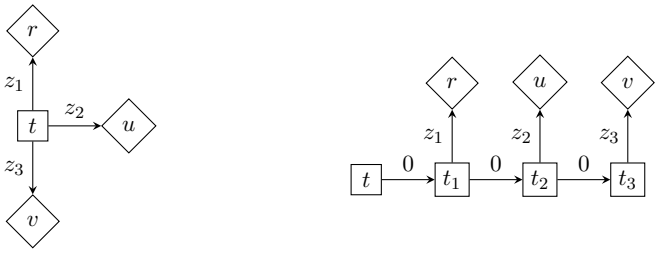


Figure 4.2: Replacing a vertex t with $\deg(t) > 2$ by a chain of vertices with degree at most two.

Now, there are three types of transitions according to the source state: from Eve's state or from Adam's state which has either one or two outgoing transitions. We construct gadgets for each case. Let's first consider the cases where Adam does not make a decision. That is moves (s, z, r) and (t, z, r), where $z \in \mathbb{Z}$, $s \in V_E$, $r \in V$, $t \in V_A$ and $\deg(t) = 1$. In the $\mathbb{Z}$-VASS game, Eve has moves $\langle\!\langle s, 4z, r \rangle\!\rangle$ and $\langle\!\langle t, 4z, r \rangle\!\rangle$, where $s, r, t \in Q_E$, respectively, and Adam has a move $\langle\!\langle \top, 0, \top \rangle\!\rangle$. The moves are depicted in Figure 4.3.

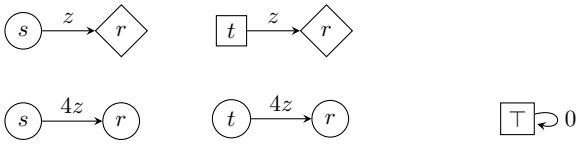


Figure 4.3: Moves in a counter reachability game (top) and the corresponding moves in the $\mathbb{Z}$-VASS game (bottom).

The final case where Adam has to make a choice is slightly more complicated. As Eve is simulating the whole graph of the counter reachability game, Adam needs to indicate to her which edge he would have picked. In the counter reachability game, the moves are $(t, y, p), (t, x, q)$, where $p, q \in V$ and $t \in V_A$ and $\deg(t) = 2$. In $\mathbb{Z}$-VASS game, Eve has a gadget with moves $\langle\!\langle t, 4y - 1, p \rangle\!\rangle$, $\langle\!\langle t, 4x + 1, q \rangle\!\rangle$, and Adam has moves $\langle\!\langle \top, 1, \top \rangle\!\rangle$ and $\langle\!\langle \top, -1, \top \rangle\!\rangle$. The moves are depicted in Figure 4.4. By multiplying all the old labels by 4, we have created extra space to store the information about which edge Eve is supposed to pick.

Finally, we need to make sure that Adam does not abuse his moves, i.e., does not indicate his choice when he should not. For this, we create a gadget similar to Adam's state transition, which Eve can enter and add ± 4 emptying the counter while at the same time cancelling whatever Adam plays. To do so, we design an emptying gadget of Eve

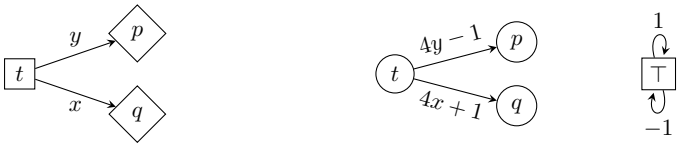


Figure 4.4: Moves in a counter reachability game (left) and the corresponding moves in the $\mathbb{Z}$-VASS (right).

consisting of one state $\bot$. The moves are $\langle\!\langle \bot, \pm 4 + 1, \bot \rangle\!\rangle$, $\langle\!\langle \bot, \pm 4 - 1, \bot \rangle\!\rangle$ and $\langle\!\langle \bot, \pm 4, \bot \rangle\!\rangle$. The emptying gadget is connected to states of Eve with moves $\langle\!\langle s, \pm 1, \bot \rangle\!\rangle$ for every state $s \in V_E$ or $s \in V_A$ and $\deg(s) = 1$, and with $\langle\!\langle t, 0, \bot \rangle\!\rangle$, where $t \in V_A$ and $\deg(t) = 2$. The control states of the players are depicted in Figure 4.5.

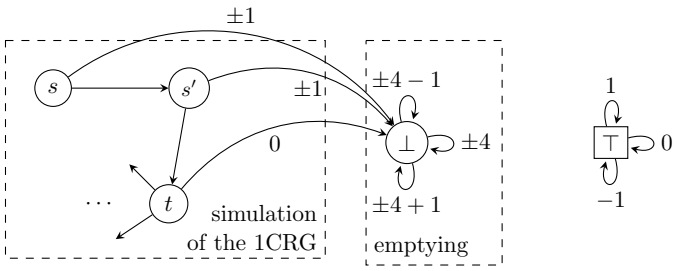


Figure 4.5: An illustration of state transitions of Eve and Adam.

Next, we consider all possible plays of Adam and Eve, and show that if the player plays incorrectly, the opponent has a winning strategy. The possible ways the game can progress are listed in Figure 4.6. First, we informally describe the incorrect moves and how the opponent can deal with them.

Adam can play incorrectly by either playing ± 1 even though Eve is not in a state where Adam has to make a decision, or by playing 0 if Eve is. In the first case, Eve can play the opposite move and move to $\bot$, after which she can counter any move Adam plays whilst emptying the counter. In the latter case, Eve moves to $\bot$ without modifying the counter and again she can empty the counter while cancelling the effect of the moves of Adam.

Eve can play incorrectly by either moving to the emptying gadgets before Adam made an incorrect move or by not making the correct decision according to what Adam has played, that is, playing $4y - 1$ after Adam played -1 or $4x + 1$ after Adam played 1. In the both cases, Adam can ensure that the counter will never be $0 \mod 4$.

First, we prove two lemmas regarding incorrect moves by Adam and prove that Eve has

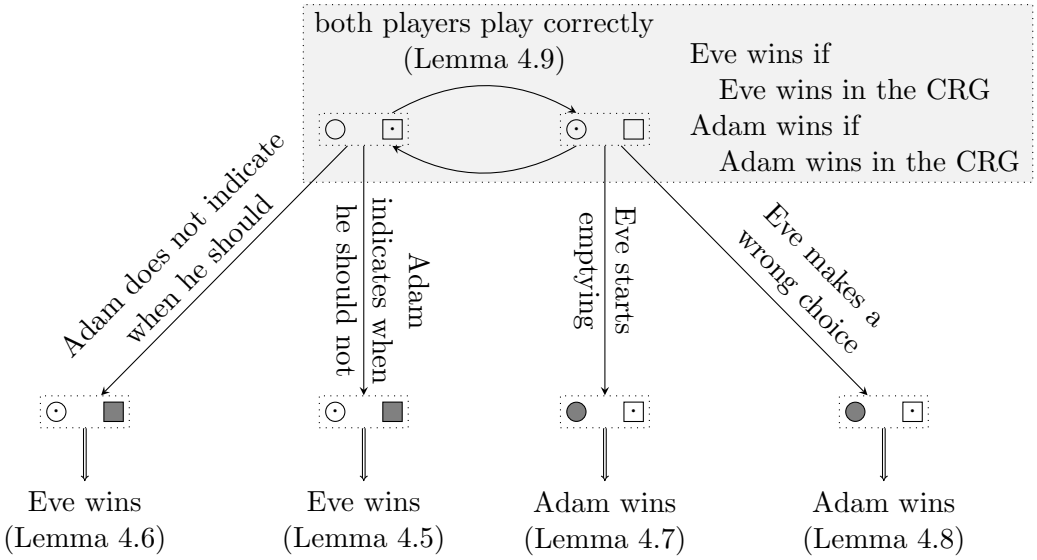


Figure 4.6: Progress of a one-dimensional $\mathbb{Z}$-VASS game.

winning strategies.

Lemma 4.5. *Let the configuration be* $[s, \dot{\top}, 4z]$*, where* $z \in \mathbb{Z}$ *and* $s \in V_E$ *or* $s \in V_A$ *and* $\deg(s) = 1$*. If Adam plays* $\langle\!\langle \top, 1, \top \rangle\!\rangle$*, then Eve has a winning strategy starting with* $\langle\!\langle s, -1, \bot \rangle\!\rangle$*. Similarly, if Adam plays* $\langle\!\langle \top, -1, \top \rangle\!\rangle$*, then Eve has a winning strategy starting with* $\langle\!\langle s, 1, \bot \rangle\!\rangle$*.*

Proof. After Adam's move, the configuration is $[\dot{s}, \top, 4z + 1]$ and after Eve's move, the configuration is $[\bot, \dot{\top}, 4z]$. After this, Eve can cancel Adam's move while emptying the counter at the same time. In the case Adam played $\langle\!\langle \top, -1, \top \rangle\!\rangle$, then Eve's winning strategy is the same after she played $\langle\!\langle s, 1, \bot \rangle\!\rangle$. □

Lemma 4.6. *Let the configuration be* $[t, \dot{\top}, 4z]$*, where* $z \in \mathbb{Z}$ *and* $t \in V_A$ *and* $\deg(t) = 2$*. If Adam plays* $\langle\!\langle \top, 0, \top \rangle\!\rangle$ *then Eve has a winning strategy starting with* $\langle\!\langle t, 0, \bot \rangle\!\rangle$*.*

Proof. After Adam's move, the configuration is $[\dot{t}, \top, 4z]$ and after Eve's move, the configuration is $\langle\!\langle \bot, \dot{\top}, 4z \rangle\!\rangle$. As in the previous lemma, Eve can empty the counter while cancelling Adam's move. □

Next, we prove a lemma, where Eve moves to her emptying gadget and prove that Adam has a winning strategy.

Lemma 4.7. *Let the configuration be $[\dot{s}, \top, 4z]$, where $z \in \mathbb{Z}$ and $s \in V_E$ or $s \in V_A$ and $\deg(s) = 1$. If Eve moves to $\perp$ with the move $\langle\!\langle s, 1, \perp \rangle\!\rangle$ or the move $\langle\!\langle s, -1, \perp \rangle\!\rangle$, then Adam has a winning strategy starting with $\langle\!\langle \top, 1, \top \rangle\!\rangle$ or $\langle\!\langle \top, -1, \top \rangle\!\rangle$ respectively.*

Proof. After Eve's move, the configuration is $[\perp, \dot{\top}, 4z \pm 1]$ and after Adam's move, the configuration is $[\dot{\perp}, \top, 4z \pm 2]$. From this moment onward, Adam can ensure that the counter is never $0 \mod 4$ after Eve's turn. Thus, Eve cannot reach counter value 0 and cannot win. □

Finally, we consider the case where Adam tells Eve his non-deterministic choice with 1 or -1, and Eve responds incorrectly by playing a move with 1 or -1, respectively, or moves to the emptying gadget.

Lemma 4.8. *Let the configuration be $[\dot{t}, \top, 4z+1]$, where $z \in \mathbb{Z}$ and $t \in V_A$ and $\deg(t) = 2$. If Eve plays the move $\langle\!\langle t, 4y+1, p \rangle\!\rangle$, then Adam has a winning strategy starting with $\langle\!\langle \top, 0, \top \rangle\!\rangle$. Symmetrically, if the configuration is $[\dot{t}, \top, 4z - 1]$ and Eve plays the move $\langle\!\langle t, 4x - 1, q \rangle\!\rangle$, then Adam has a winning strategy. If the configuration is $[\dot{t}, \top, 4z \pm 1]$ and Eve plays the move $\langle\!\langle t, 0, \perp \rangle\!\rangle$, then Adam has a winning strategy.*

Proof. In the first case, after Eve's move, the configuration is $[p, \dot{\top}, 4(z + y) + 2]$ and Adam with his moves can ensure that the counter is not $0 \mod 4$. That is, Eve cannot reach counter value 0 and thus cannot win. Symmetrically, if after Eve's move the configuration is $[p, \dot{\top}, 4(z + x) - 2]$, then Adam can ensure that the counter is not $0 \mod 4$ and Eve cannot win.

In the third case, after Eve's move, the configuration is either $[\perp, \dot{\top}, 4z+1]$ or $[\perp, \dot{\top}, 4z-1]$. By playing $\langle\!\langle \top, 1, \top \rangle\!\rangle$ in the first case and $\langle\!\langle \top, -1, \top \rangle\!\rangle$ in the second, Adam can ensure that the counter is not $0 \mod 4$ as in the previous cases. □

Next, we prove that if both players play correctly, the winner is the same as in the one-dimensional counter reachability game.

Lemma 4.9. *If in the one-dimensional $\mathbb{Z}$-VASS game constructed previously Eve plays*

- *the move $\langle\!\langle t, 1, p \rangle\!\rangle$ if the configuration is $[\dot{t}, \top, 4z - 1]$ for some $z \in \mathbb{Z}$ and $t \in V_E$ and $\deg(t) = 2$,*
- *the move $\langle\!\langle t, -1, p \rangle\!\rangle$ if the configuration is $[\dot{t}, \top, 4z + 1]$ for some $z \in \mathbb{Z}$ and $t \in V_E$ and $\deg(t) = 2$*

and never moves to $\bot$, and Adam plays

- *the move $\langle\!\langle \top, 0, \top \rangle\!\rangle$ if the configuration is $[s, \dot{\top}, 4z]$, for some $z \in \mathbb{Z}$ and $s \in V_E$ or $s \in V_A$ and $\deg(s) = 1$,*
- *a move $\langle\!\langle \top, -1, \top \rangle\!\rangle$ or $\langle\!\langle \top, 1, \top \rangle\!\rangle$ if the configuration is $[t, \dot{\top}, 4z]$, for some $z \in \mathbb{Z}$ and $t \in V_A$ and $\deg(t) = 2$,*

then Eve has a winning strategy if and only if she has a winning strategy in the one-dimensional counter reachability game.

Proof. It is easy to see that these moves simulate the counter reachability game and that Eve has a winning strategy to reach the configuration $[f, 0]$ of the counter reachability game if and only if she has a winning strategy to reach the configuration $[\dot{f}, \top, 0]$ in the $\mathbb{Z}$-VASS game. □

We are ready to prove the main theorem.

Theorem 4.4. *The one-dimensional $\mathbb{Z}$-VASS games are* EXPSPACE*-complete.*

Proof. By Lemma 4.3, deciding the winner is in EXPSPACE. It remains to be proven that it is also EXPSPACE-hard. Let $((V, F), c_0)$ be a one-dimensional counter reachability game. Let (A, E, c_0') be the $\mathbb{Z}$-VASS game constructed from $((V, F), c_0)$. Assume first that Eve has a winning strategy in $((V, F), c_0)$. Now, Eve's winning strategy in the $\mathbb{Z}$-VASS game (A, E, c_0') is to play according to the winning strategy of $((V, F), c_0)$ if the configuration is $[\dot{s}, \top, 4z]$, where $s \in V_E$ or $s \in V_A$ and $\deg(s) = 1$. If the configuration is $[\dot{t}, \top, 4z - 1]$ or $[\dot{t}, \top, 4z + 1]$, where $t \in V_A, \deg(t) = 2$, then Eve plays moves $\langle\!\langle t, 4x + 1, q \rangle\!\rangle$ or $\langle\!\langle t, 4y - 1, p \rangle\!\rangle$, respectively. This is a winning strategy by Lemma 4.9. If the configuration is $[\dot{s}, \top, 4z \pm 1]$, where $s \in V_E$ or $s \in V_E$ and $\deg(s) = 1$, then Eve has a winning strategy by Lemma 4.5. If the configuration is $[\dot{t}, \top, 4z]$, where $t \in V_A$ and $\deg(t) = 2$, then Eve has a winning strategy by Lemma 4.6.

Assume then, towards a contradiction, that Adam has a winning strategy in $((V, F), c_0)$ and Eve has a winning strategy in (A, E, c_0'). By Lemma 4.9, Adam has a winning strategy if the players simulate the counter reachability game correctly. That is, Eve has to, at some point, either move to the emptying gadget, or play $\langle\!\langle (t, 4x \pm 1, s) \rangle\!\rangle$ when the configuration is $[\dot{t}, \top, 4z \mp 1]$. By Lemma 4.7 and Lemma 4.8, Adam has winning strategies for both cases. As we have analysed all the possible moves of Eve, we have shown that Eve does not have a winning strategy. □

This result clearly extends to any dimension d. In other words, d-dimensional counter reachability games and $\mathbb{Z}^d$-VASS games are equivalent.

Corollary 4.10. *For a d-dimensional counter reachability game, there exists a $\mathbb{Z}^d$-VASS game such that Eve has a winning strategy in the counter reachability game if and only if she has a winning strategy in the $\mathbb{Z}^d$-VASS game.*

4.2 Flat $\mathbb{Z}$-VASS games in dimension one

There is an interesting complexity difference between games of Theorem 4.2 and Theorem 4.4. When the (stateless) $\mathbb{Z}$-VAS game is extended by allowing Eve to have an internal state structure and keeping Adam stateless, the complexity of deciding the winner increases from EXPTIME to EXPSPACE. In this section we study a natural dual question — does keeping Eve stateless and allowing Adam to have an internal structure result in a similar increase? We study this question by considering $\mathbb{Z}$-VASS games where Eve is stateless and Adam's states are flat (i.e., the underlying graph is directed acyclic graph with self-loops), called flat $\mathbb{Z}$-VASS games. The state structure of a flat $\mathbb{Z}$-VASS is depicted in Figure 4.7. The main result of the section is that deciding the winner in the flat $\mathbb{Z}$-VASS games is in EXPTIME. Note, that as the stateless $\mathbb{Z}$-VASS games are also flat $\mathbb{Z}$-VASS, we have inherited EXPTIME-hard lower-bound.

Remark 4.11. Let (A, E, z_0) be a $\mathbb{Z}$-VAS game. If the winning set is nontrivial, then it is either $d\mathbb{Z}$, for some integer d, or $R \cup U \subseteq \mathbb{Z}^+$, where $U = \{x \in d\mathbb{Z} \mid x > b\}$ and R consists of the winning values on the finite arena $[0, b]$, or $R' \cup U' \subseteq \mathbb{Z}^-$, where $U' = \{x \in d\mathbb{Z} \mid x < b\}$ and R' consists of the winning values on the finite arena $[b, 0]$. Also note, that the initial value z_0 is not important as all winning values are computed.

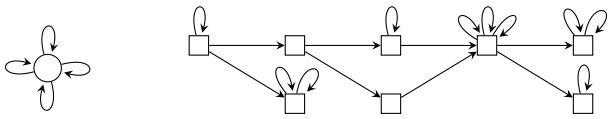


Figure 4.7: An example of a flat $\mathbb{Z}$-VASS game.

Before considering arbitrary flat graphs of Adam, we consider a simpler case where there are three types Adam's moves: self-loops in state t_0, self-loops in state t_1 and transitions from t_0 to t_1.

The idea is that there are two stateless $\mathbb{Z}$-VAS games when moves are restricted to self-loops and additional moves connecting the games. The algorithm of [9] not only computes whether the given initial value z_0 is winning for Eve, but it computes the set of all winning values. We can use the algorithm to compute winning sets for both games and then connect the two games using the transitions between t_0 and t_1.

Example 4.12. Consider a one-dimensional flat $\mathbb{Z}$-VASS where Eve's moves are

$$\{\langle\!\langle s, -3, s\rangle\!\rangle, \langle\!\langle s, -6, s\rangle\!\rangle, \langle\!\langle s, -7, s\rangle\!\rangle, \langle\!\langle s, -8, s\rangle\!\rangle\}$$

and Adam's moves are $\{\langle\!\langle t_0, -3, t_0\rangle\!\rangle, \langle\!\langle t_0, -6, t_0\rangle\!\rangle, \langle\!\langle t_0, 0, t_1\rangle\!\rangle, \langle\!\langle t_1, -7, t_1\rangle\!\rangle, \langle\!\langle t_1, -8, t_1\rangle\!\rangle\}$. It is easy to compute the winning sets for games restricting to t_0 and t_1: $W_0 = 9\mathbb{Z}^+$ and $W_1 = \{0, 14, 15, 25, 28, 29, 30, 39, 40, 41, 42, 43, 44, 45\} \cup \{x \mid x \geq 50\}$, respectively.

We notice that, for example, 9 is not a winning value in the flat $\mathbb{Z}$-VASS game. Indeed, while by staying in t_0, Adam loses, if he instead moves to t_1, then after Eve's turn the counter will be $1, 2, 3$ or 6. None of these is a winning value when restricting to t_1. On the other hand, all the other winning values, that is $9k$, where $k > 1$, can reach 0 only by reaching 9 first. That is, Eve does not have any winning values. This is illustrated in Figure 4.8.

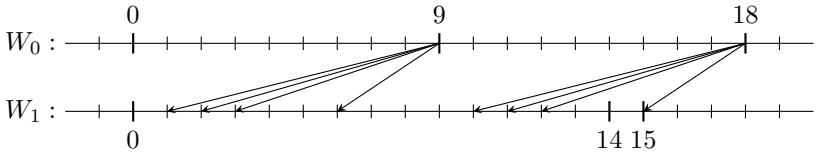


Figure 4.8: An illustration of connecting winning sets in a flat $\mathbb{Z}$-VASS game.

For this special case, there are three steps needed to compute the winning set of the game.

- Compute the winning sets of restricted games, W_0 and W_1.
- Compute the forbidden values F in W_0, that is, all the values in W_0 from which there exists a move $\langle\!\langle t_0, z, t_1\rangle\!\rangle$ of Adam such that for any move $\langle\!\langle s, x, s\rangle\!\rangle$ of Eve, the resulting value is not in W_1.
- Finally, check whether values of F are avoidable in W_0. That is, whether there exists a winning strategy from the initial value z_0 to 0 that does not visit any values of F.

The first step can be done in EXPTIME using the algorithm for $\mathbb{Z}$-VAS games. The second and third steps require some additional considerations as the sets are potentially infinite. In the game of the previous example, if the initial value is z_0, then it is not important to check which forbidden values larger than z_0 are avoidable and which are not. On the other hand, it is easy to see that in general case, it is not a simple matter of discarding larger values than z_0 (assuming that z_0 is positive). By Remark 4.11, the winning set of a $\mathbb{Z}^d$-VAS game has a particular structure. In a similar manner, the set of forbidden values constructed from two winning sets have some structure which allows us to compute whether the values are avoidable. Now, there are two sets of forbidden values, one resulting from the finite set of R, $F_{\text{fin}} = \{f_1, \ldots, f_k\}$, and a regular but infinite set of values resulting from U, F_{inf}. Even though, F_{inf} is infinite, it is semi-linear (and in fact linear when Adam has two states). We can extract a finite set of forbidden values, F', such that $F_{\text{inf}} = \bigcup_{i=0}^{\infty} F' + i\ell$ for some $\ell \in \mathbb{Z}$. Now, we have two finite sets of forbidden values for which it is easy to check whether the values are avoidable. We can use the attractor construction found in Chapter 2 of [75] which solves the game in polynomial time. In our example, $F = \{9, 27\}$ and 9 is reachable only from one winning value, namely 18. On the other hand, 9 is the only winning value reachable from 18, so 9 is not avoidable.

Lemma 4.13. *Let (A_0, E, z_0) and (A_1, E, z_0) be two $\mathbb{Z}$-VAS games and T the set of labels of Adam's moves connecting the two games. Let W_0 and W_1 be their respective winning sets. The set $F = \{x \in W_0 \mid \exists z \in T\, \forall \langle\!\langle s, y, s \rangle\!\rangle \in E : x + z + y \notin W_1\}$ can be computed in polynomial time.*

Proof. There are several cases to consider. First, we have two trivial cases when one of the winning sets is trivial, i.e., $\{0\}$. If the winning set W_0 is trivial, then $F = \{0\}$. If the winning set W_1 is trivial, then $F = W_0$. Another obvious case is when the winning set $W_0 \subseteq \mathbb{Z}^+$ and $W_1 \subseteq \mathbb{Z}^-$ (or the symmetric situation), then there are only finitely many points in W_0 from which it is possible to reach W_1. Thus $F = W_0 \setminus X$, where X is a finite subset of $(0, a]$ for some a bounded by $\max(E) + \min(T)$. There remain four cases.

1. $W_0 = d\mathbb{Z}$ and $W_1 = d'\mathbb{Z}$, or
2. $W_0 = R \cup U$ and $W_1 = d'\mathbb{Z}$, or
3. $W_0 = R \cup U$ and $W_1 = R' \cup U'$, or
4. $W_0 = d\mathbb{Z}$ and $W_1 = R' \cup U'$.

Recall that R and R' are finite and $U = \{x \in d\mathbb{Z} \mid x > b\}$. Consider the first case. Let $\ell = \operatorname{lcm}(d, d')$. We can partition the integer line into intervals of length ℓ and effectively compute all the forbidden values F' in the interval. Clearly, the forbidden values in one interval, will be also forbidden in the other intervals. The set of all forbidden values is $F = \{f + \ell i \mid f \in F', i \in \mathbb{Z}\}$.

The next case can be divided into two parts, first finding forbidden values in R and then in U. Finding the forbidden values in R is easy as there are only finitely many possible values. Finding the forbidden values in U can be done as for the first case. The third and fourth cases are done similarly but now we also have to take the finite set R' into account. In all three cases, the set of forbidden values is $F = \{f_1 \ldots, f_k\} \cup \left[\bigcup_{i=0}^{\infty} F' + i\ell\right]$, where $|F'| < \infty$ and $f > f_j$ for all indexes j and $f \in F'$. □

Be the previous lemma, we can compute the forbidden values in polynomial time. It remains to be shown that we can decide the winner in game, where we take the forbidden values into account. To this end, we look at the winning values that remain winning if we avoid the forbidden values.

Lemma 4.14. *Let (A_0, E, z_0) be a $\mathbb{Z}$-VAS game and $W_0 \subseteq \mathbb{Z}^+$ its winning set. Let $F_{fin} \subseteq (0, a] \subseteq W_0$ be a subset of forbidden values in W_0 and $F_{inf} \subseteq (a, b] \subseteq W_0$ such that the set of all forbidden values is $F_{fin} \cup \left[\bigcup_{i=0}^{\infty} F_{inf} + i(b-a)\right]$. There exists a finite set X such that $F_{fin} \cup F_{inf} \subseteq X \subseteq W_0$ and we can compute the values of X avoiding the values of F in polynomial time. The symmetrical claim holds if the winning set consists of only negative values.*

Proof. Let $m = \min(A_0)$ and $M = \max(A_0)$ be the smallest and the largest moves of Adam. Let $X = (m, b + (b-a) + M]$. Clearly $F_{\text{fin}} \cup F_{\text{inf}} \subseteq X$. We can construct a reachability game on a finite arena X by having two copies of the interval X, one for Eve and one for Adam. We connect integers in Eve's (Adam's) interval to integers in Adam's (Eve's) interval corresponding to her (his) moves.

The interval X can be partitioned into three parts, $(-m, a]$, $(a, b]$ and $(b, b+(b-a)+M]$. Intuitively, the first interval $(-m, a]$ corresponds to F_{fin}, the second interval $(a, b]$ to F_{inf} and the final interval to the set $\bigcup_{i=1}^{\infty} F_{\text{inf}} + i(b-a)$. As the finite interval corresponds to several sets, Eve can also move from $x \in (b+m, 2b-a+M]$ to y if there exists a move from $x + (b-a)$ to y. Additionally, if $f \in F_{\text{fin}} \cup F_{\text{inf}} \cup F_{\text{inf}} + b - a$, then Adam can move to the sink state $\top$ which is losing for Eve. Finally, there exists an edge from a state x if the owner of the state has a move y in the $\mathbb{Z}$-VAS game such that $x + y < 0$.

More formally, Adam has states $Q_A = \{\square \times [0, 2b - a]\} \cup \{\top\}$ and Eve has states $Q_E = \{\bigcirc \times [m, 2b - a + M]\}$. The transitions of the game are

$$\begin{aligned} T = &\{((\square, x), (\bigcirc, y)) \in Q_A \times Q_E \mid y - x \in A\} \\ &\cup \{((\bigcirc, x), (\square, y)) \in Q_E \times Q_A \mid y - x \in E\} \\ &\cup \{((\bigcirc, x), (\square, y)) \in Q_E \times Q_A \mid y - (x + b - a) \in E, x \in (b + m, 2b - a + M]\} \\ &\cup \{((\square, x), \top) \mid x \in F_{\text{fin}} \cup F_{\text{inf}} \cup F_{\text{inf}} + b - a\} \\ &\cup \{((\bigcirc, x), \top) \in Q_E \times Q_A \mid \exists e \in E, x + e < 0\} \cup \{(\top, \top)\}. \end{aligned}$$

Eve wins the game if she can reach $(\square, 0)$. The winning values of this game can be computed using the attractor construction in polynomial time [75]. □

Example 4.15. Let $(\{\langle\!\langle t, -1, t\rangle\!\rangle\}), (\{\langle\!\langle s, -1, s\rangle\!\rangle, \langle\!\langle s, -2, s\rangle\!\rangle\})$ be a $\mathbb{Z}$-VAS game. Let $F = \{3\} \cup \{x \in \mathbb{Z}^+ \mid x \equiv 2 \mod 3, x > 2\}$ be the set of forbidden values. By Lemma 4.14 we can construct a reachability game G depicted in Figure 4.9.

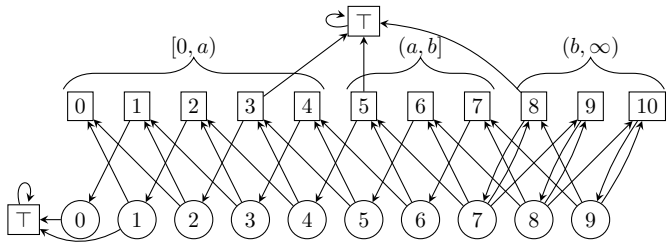


Figure 4.9: A reachability game on finite arena constructed from a $\mathbb{Z}$-VAS game and a set of forbidden values.

Now we are ready to extend Adam's state structure to flat graphs. The algorithm is essentially the same as the one described previously. We utilise the topological sorting to remove forbidden points from the winning sets starting from the end of the graph using Lemma 4.13. Then we construct the set of avoidable values using Lemma 4.14.

Theorem 4.16. *One-dimensional flat* $\mathbb{Z}$-VASS *games are* EXPTIME*-complete.*

Proof. Let (A, E, c_0) be a flat $\mathbb{Z}$-VASS, where Adam has k states, $t_1, \ldots, t_k$, such that $\langle\!\langle t_i, z, t_j\rangle\!\rangle \in A$ only if $i \leq j$. Denote by $A_i = \{\langle\!\langle t_i, z, t_i\rangle\!\rangle \in A\}$. Using the algorithm of [9], we compute the winning set for each pair (A_i, E). Then, starting from k, we compute sets

of forbidden values using Lemma 4.13. After computing the forbidden values, we compute the avoidable values using Lemma 4.14. Finally, we update the sets of winning values using the forbidden and avoidable values. □

It is natural to consider the dual of Theorem 4.16. That is, what is the complexity of $\mathbb{Z}$-VASS games where Adam is stateless and Eve has flat states? Following the result of [110] on flatness of vector addition systems with states, we analyse whether the corresponding games are also flat. In [110], it was proven that one-dimensional VASS are flat. As VASS games of [1, 29] are also VASS, so VASS games are also flat. The reductions in [138] and [88] do not create nested loops, so the games remain flat. The same thing applies to our reduction from counter reachability games to $\mathbb{Z}$-VASS games. Thus, we have a following result on $\mathbb{Z}$-VASS games where Adam is stateless and Eve has flat states:

Corollary 4.17. *One-dimensional* $\mathbb{Z}$-VASS *games, where Adam is stateless and Eve has flat states, are* EXPSPACE*-complete.*

4.3 VAS games in dimension one

In this section, we consider VAS games, i.e., where the arena is limited to the positive half line $\mathbb{Z}^+$.

Remark 4.18. In a d-dimensional VAS game, if the configuration is $\mathbf{x}$ and all moves of a player result in a configuration, where at least one component is negative, then that player loses. In other words, forcing another player into a deadlock is a winning condition.

Theorem 4.19. *One-dimensional* VAS *games are* EXPTIME*-complete.*

Proof. Consider, the algorithm of [9, 137] that computes the winning set for Eve in a given one-dimensional $\mathbb{Z}$-VAS game. It can be modified to take the VAS semantics into account. The algorithm computes the winning set for Eve, which consists of two sets. See Remark 4.11 for more detailed description. The first set is the infinite set that is similar to the solution of the Frobenius problem. That is, integers sufficiently far from the origin are winning for Eve. The second set is a finite set consisting of winning values that are winning for Eve but are not "sufficiently" far from the origin. It is easy to see that the former set is not affected by the VAS semantics and in the latter set VAS semantics can be easily implemented as the set is finite. That is, deciding which player has a winning strategy in VAS games is decidable in EXPTIME.

The proof of EXPTIME-hardness of one-dimensional $\mathbb{Z}$-VAS games [9,137] applies as such. In the proof, Adam's moves are positive, Eve's moves are negative. Moreover, $a + e < 0$ for every $a \in A$ and $e \in E$. That is, only Eve can reach a deadlock and the game is played for finitely many rounds until either Eve wins by reaching 0 or Eve reaches a deadlock and Adam wins.

We have proved that one-dimensional VAS games are EXPTIME-complete. □

4.4 Concluding remarks and open problems

In this chapter, we considered several one-dimensional games. The most important result of the chapter is showing that deciding which player has a winning strategy in $\mathbb{Z}$-VASS games is EXPSPACE-complete problem. In our construction Eve had states, while Adam was stateless. Motivated by this, we considered games where Adam had states and Eve was stateless. When limiting Adam's state structure to flat automata, we showed that the games are EXPTIME-complete. Furthermore, but analysing the construction of previous results on VASS and CRG games, we showed that the problem is EXPSPACE-complete when Eve's state structure is flat. The tight complexity of $\mathbb{Z}$-VASS games where Adam has an arbitrary state structure and Eve is stateless, remains open.

Chapter 5

Two-dimensional $\mathbb{Z}$-VAS games

In the previous chapters, we proved that most of Attacker-Defender games are undecidable when moves are multidimensional objectives (such as matrices and pairs of words) and decidable when moves are one-dimensional (such as binary words and integers). The decidability status of multidimensional games, where moves are integers, has been studied in [1, 2, 29], where the authors proved that in games with VASS semantics, the problem is undecidable. That is, in games played on a single graph with vertices partitioned between players and $\mathbb{N}^d$ being the vector space. Later, in [136, 138], the model was extended to counter reachability games, where the vector space is $\mathbb{Z}^d$. Doyen and Rabinovich [62] proposed two simple game scenarios with open decidability status, that they called robot games and robot games with states.

In robot games, or $\mathbb{Z}$-VAS games in our terminology, the authors claimed, based on personal communications, that the problem is decidable in dimension one, which was later published as [9] and that the problem is undecidable starting from dimension nine. On the other hand, the robot games with states, or $\mathbb{Z}$-VASS games as we call them, were claimed to be undecidable starting from dimension three. Moreover, they claimed that games with states of dimension d can be reduced to games without states of dimension $d+6$. Unfortunately, the latter three results were never published. Later, significant progress was done by Reichert, who proved in his thesis [137] that $\mathbb{Z}^2$-VASS games and $\mathbb{Z}^3$-VAS games are undecidable.

In the chapter, we solve the open problem of [62] of deciding the winner of $\mathbb{Z}^d$-VAS games for dimension $d = 2$. We close the gap by showing that it is undecidable to check which of the players has a winning strategy in a two-dimensional $\mathbb{Z}$-VAS game, i.e., in a

very restricted fragment of counter reachability games with stateless players playing on integer grid $\mathbb{Z}^2$. Before studying the stateless games, we consider two-dimensional games on integer vector addition systems with states, to illustrate the proof better.

The undecidablity of $\mathbb{Z}^2$-VASS games was proved already in [137] and we first reiterate the proof with different notation. Then we present our proof of undecidability of $\mathbb{Z}^2$-VAS games that incorporates the original construction from the proof of three-dimensional games, together with new techniques that allow us to encode one of the dimensions into the other two.

The basis of our proofs are two-counter Minsky machines, for which the halting problem is undecidable. For a two-counter machine, we construct a game where Eve has to simulate the machine and Adam verifies that Eve does not cheat. The intuition is that the counters of the machine are multiplied by constants and represented by two-dimensional vectors. Additionally, the states of the machine are encoded in the least significant digits of the vectors. We analyse all the possible deviations from simulating the counter machine and show that the opponent has a winning strategy in that case. The biggest challenge is to ensure that all possible ways to cheat can be caught without introducing new ways to cheat for the other player.

We prove the main theorem by considering the undecidable problem of determining whether a 2CM $\mathcal{M}$ reaches a configuration where both counters are zero. In Section 5.1, we construct a $\mathbb{Z}^2$-VASS game that follows the computation of $\mathcal{M}$. To simulate zero checks present in two-counter machines, Adam has a move allowing him to check whether or not a counter is positive. This check leads, deterministically, either to Adam's victory with a correct guess or to his loss otherwise. In the second section, we map the states and state transitions into integers and embed them into the least significant digits in vectors of a two-dimensional $\mathbb{Z}$-VAS game. Our proof uses two successive reductions making the proof shorter and more intuitive in contrast to a direct reduction from 2CM that would lead to a longer proof with significantly more cases to consider.

Known results on $\mathbb{Z}^d$-VAS games in different dimensions are summarised in Table 5.1. Apart from the solution of the open problem, the main contribution of the chapter is a collection of new, original encodings and constructions that allow simulating zero-checks and state space of a universal machine within a minimalistic two-dimensional system of two non-deterministic stateless players.

We conclude the chapter with considerations of two-dimensional VAS games and show that, similarly to Section 4.3, restricting the arena to positive quadrant does not affect the

Game \ Dimension	1	2	3
$\mathbb{Z}^d$-VAS	EXPTIME-complete [9]	undecidable (Corollary 5.13)	undecidable [137]
VAS	EXPTIME-complete (Theorem 4.19)	undecidable (Theorem 5.14)	—

Table 5.1: The results on complexity of deciding whether Eve has a winning strategy in $\mathbb{Z}^d$-VAS and VAS games.

complexity. That is, the problem remains undecidable.

5.1 $\mathbb{Z}$-VASS games in two dimensions

In this section, we prove that the decision problem for $\mathbb{Z}^2$-VASS games is unsolvable. By Corollary 4.10, the undecidability of $\mathbb{Z}^2$-VASS games follows from the undecidability of counter reachability games [138]. We use a different reduction that makes the reduction in Section 5.2 easier to follow. This construction was first presented in [137].

We show that for each two-counter machine, there exists a corresponding $\mathbb{Z}^2$-VASS game where Eve has a winning strategy if and only if the machine reaches a configuration where both counters are zero. The simulation is rather straightforward apart from zero-checks of the machine. To simulate zero checks present in two-counter machines, we use two-player dynamics by giving Adam a move allowing him to check whether the counter is positive or not.

Theorem 5.1. *Let (Q, T, s_0) be a two-counter machine. There exists a two-dimensional $\mathbb{Z}$-VASS game (A, E, c_0) where Eve has a winning strategy if and only if (Q, T, s_0) reaches a configuration in $Q \times \{(0, 0)\}$.*

The idea is that in the $\mathbb{Z}^2$-VASS, Eve simulates the computation of the 2CM while Adam does not interfere with the computation. If Eve deviates from the computation, then Adam has a winning strategy from that point on. On the other hand, if Adam intervenes in a faithful simulation, then Eve has a winning strategy.

Essentially, there are four ways the game can progress. These ways are depicted in Figure 5.1. Three of the outcomes have a predetermined winner which does not depend on the 2CM. In the last case where Eve correctly simulates the 2CM and Adam does not

interfere (plays only a 0-MOVE), the winner depends on whether the 2CM reaches $[s, (0, 0)]$ for some $s \in Q$ or not.

- If Eve's move corresponds to the SIMULATION of the 2CM and Adam replies with a 0-MOVE (a move that does not modify the counters), then iteratively applying only this turn-based interaction, Eve wins if and only if the 2CM reaches $[s, (0, 0)]$ for some $s \in Q$ (Lemma 5.2).
- If Eve's move incorrectly simulates the 2CM, then Adam has a winning strategy from this moment on, starting with a POSITIVITY CHECK that makes Eve's target unreachable (Lemma 5.3).
- If Adam plays his POSITIVITY CHECK following a correct simulating move of Eve, then Eve has a winning strategy from this moment on, starting with an EMPTYING MOVE, allowing Eve to empty both counters and reach $(0, 0)$ (Lemma 5.4).
- Finally, if Eve plays an EMPTYING MOVE instead of a SIMULATING MOVE, in that case Adam has a winning strategy starting by playing his 0-MOVE (Lemma 5.5).

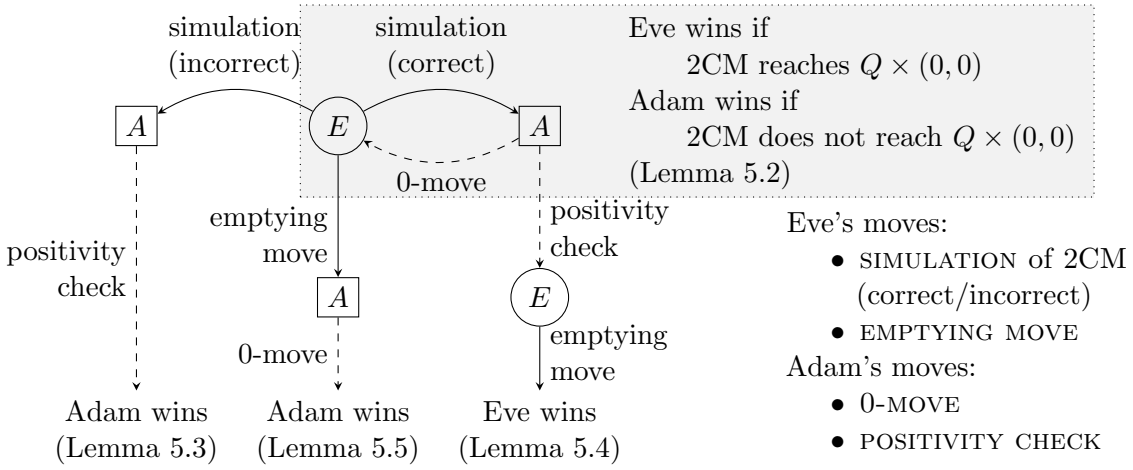


Figure 5.1: Progress of a $\mathbb{Z}^2$-VASS game.

Before presenting the detailed constructions of Eve's and Adam's state spaces, we consider a simple modification to a 2CM, making it non-deterministic. For any 2CM (Q, T, s_0), we construct a 2CM (Q', T', s_0'), where Q' is Q with additional information on positivity of the both counters and T' is like T with guards ensuring that the extra

information in states of Q' correspond to the actual values of the counters. We denote the states of Q' by s_{ab}, where $s \in Q$ and $a, b \in \{0, +\}$ are *flags* indicating whether the value of a counter is positive or equal to 0, i.e., a (b) is $+$ if the first (second) counter is positive or 0 if the counter is zero. The transition set T' consists of the following sets

$$\{\langle s_{ab}, c_{1}{+}{+}, t_{+b}\rangle \mid \langle s, c_{1}{+}{+}, t\rangle \in T, a, b \in \{0,+\}\}, \{\langle s_{ab}, c_{2}{+}{+}, t_{a+}\rangle \mid \langle s, c_{2}{+}{+}, t\rangle \in T, a, b \in \{0,+\}\},$$
$$\{\langle s_{+b}, c_{1}{-}{-}, t_{ab}\rangle \mid \langle s, c_{1}{-}{-}, t\rangle \in T, a, b \in \{0,+\}\}, \{\langle s_{a+}, c_{2}{-}{-}, t_{ab}\rangle \mid \langle s, c_{2}{-}{-}, t\rangle \in T, a, b \in \{0,+\}\},$$
$$\{\langle s_{0b}, c_{1}{=}{=}0, t_{0b}\rangle \mid \langle s, c_{1}{=}{=}0, t\rangle \in T, b \in \{0,+\}\}, \{\langle s_{a0}, c_{2}{=}{=}0, t_{a0}\rangle \mid \langle s, c_{2}{=}{=}0, t\rangle \in T, a \in \{0,+\}\}.$$

Now, after decrementing counters from a state with $+$ flag, a state will be changed to a state with $+$ or 0 flag depending on the current counter value.

counter value	flag		flag	
$c_i > 1$	$+$	$\to$	$+$	**correct flag**
$c_i > 1$	$+$	$\to$	0	wrong flag
$c_i = 1$	$+$	$\to$	$+$	wrong flag
$c_i = 1$	$+$	$\to$	0	**correct flag**

At the moment, we assume that the machine moves to a state with the correct flag (correct simulation) and does not move to the incorrect flag (incorrect simulation). Later in the $\mathbb{Z}^2$-VASS game, Adam will act as guards (i.e., checks whether $c_i > 1$ or $c_i = 1$) using his POSITIVITY CHECK if Eve picks a wrong transition resulting in a state with the wrong flag.

Now we present the moves of the players. Eve's states are the states of Q', corresponding to the simulation of the 2CM, together with emptying states $\{\top_{00}, \top_{+0}, \top_{0+}, \top_{++}\}$, associated with EMPTYING MOVES. The moves of Eve correspond to transitions in T' where incrementing and decrementing of the first counter is by 4 rather than by 1. We call these moves SIMULATING MOVES.

Transition with c_1	Eve's move
$\langle s, c_1{+}{+}, t\rangle$	$\langle\!\langle s, (4,0), t\rangle\!\rangle$
$\langle s, c_1{-}{-}, t\rangle$	$\langle\!\langle s, (-4,0), t\rangle\!\rangle$
$\langle s, c_1{=}{=}0, t\rangle$	$\langle\!\langle s, (0,0), t\rangle\!\rangle$

Transition with c_2	Eve's move
$\langle s, c_2{+}{+}, t\rangle$	$\langle\!\langle s, (0,1), t\rangle\!\rangle$
$\langle s, c_2{-}{-}, t\rangle$	$\langle\!\langle s, (0,-1), t\rangle\!\rangle$
$\langle s, c_2{=}{=}0, t\rangle$	$\langle\!\langle s, (0,0), t\rangle\!\rangle$

The other type of moves, EMPTYING MOVES, are related to the new states and are used to empty the counters. Note that there is a hierarchy in the emptying states — Eve cannot move from a state with 0 to a state with $+$. Let us define the emptying partition of Eve's automaton where for every possible move of Adam there is a cancelling move with additional decrementing of the counters eventually leading to the sink state $\top_{00}$.

- $\{\langle\!\langle \top_{++}, (-4-e,-1), t\rangle\!\rangle \mid e \in \{0,1\}, t \in \{\top_{++}, \top_{+0}, \top_{0+}, \top_{00}\}\}$;
- $\{\langle\!\langle \top_{+0}, (-4-e,0), t\rangle\!\rangle \mid e \in \{0,1\}, t \in \{\top_{+0}, \top_{00}\}\}$;
- $\{\langle\!\langle \top_{0+}, (-e,-1), t\rangle\!\rangle \mid e \in \{0,1\}, t \in \{\top_{0+}, \top_{00}\}\}$;
- $\{\langle\!\langle \top_{00}, (-e,0), \top_{00}\rangle\!\rangle \mid e \in \{0,1\}\}$.

Finally, we define transitions connecting the simulating partition of Eve's automaton with the emptying partition. For each state $s_{ab} \in Q'$, Eve has a transition $\langle\!\langle s_{ab}, (-1,0), \top_{ab}\rangle\!\rangle$.

Adam is stateless, i.e., he has one state and his moves are self-loops. There are two types of moves: the 0-MOVE, $(0,0)$, with which Adam agrees that Eve simulated the 2CM correctly and the POSITIVITY CHECK, $(1,0)$, with which Adam checks whether a flag matches the counter (i.e., Eve simulated incorrectly). Control states of the players are depicted in Figure 5.2.

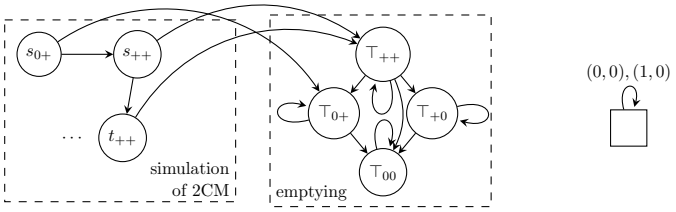


Figure 5.2: An illustration of state transitions of Eve and Adam.

Since Adam is stateless, we simplify the notation by writing the configuration as $[s, \mathbf{x}]$, i.e., we omit Adam's state. To avoid Eve winning trivially every play in the $\mathbb{Z}^2$-VASS game, we do not use $[s'_{00}, (0,0))]$ as the initial configuration, but instead consider the configuration that is reached in (Q', T') after one step of the run of the machine. We write the configuration after one step as $[s_{\bar{a}\bar{b}}, (y,z)]$ and we define $\bar{a} = +, \bar{b} = 0$ if $y = 1$ and $\bar{a} = 0, \bar{b} = +$ if $y = 0$. The initial configuration c_0 in the $\mathbb{Z}^2$-VASS game is then $[s_{\bar{a}\bar{b}}, (4y, z)]$. The effect of simulating moves, emptying moves and positivity check modulo four is depicted in Figure 5.3.

Next, we prove which player has a winning strategy in the scenarios presented previously.

Lemma 5.2. *In a sequence where Adam plays only the* 0-MOVE *and Eve plays only correct* SIMULATING MOVES*, Adam wins if the two-counter machine does not reach a configuration with zeros in both counters and Eve wins otherwise.*

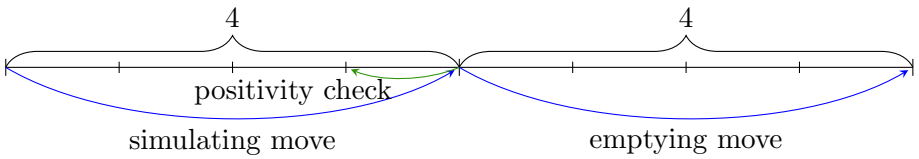


Figure 5.3: An illustration of changes in an interval when simulating or emptying moves of Eve or positivity check of Adam are applied.

Proof. It easy to see that correct moves of Eve simulate the 2CM and that a configuration $[s, (0, 0)]$ of the 2CM is reachable if and only if it is reachable in the $\mathbb{Z}^2$-VASS game. □

Lemma 5.3. *If Eve plays an incorrect move, i.e., after her turn a flag does not match the counter value (i.e., the flag is + while the counter is 0 or vice versa), then Adam has a winning strategy starting with the* POSITIVITY CHECK.

Proof. Assume that Eve made a mistake regarding the positivity of the first counter. As noted previously, there are two ways she can make a mistake. Either the configuration is $[s_{0b}, (4x, y)]$, where $x \geq 1$ or $[s_{+b}, (0, y)]$. In both cases Adam plays his POSITIVITY CHECK which changes the parity of the first counter. That is, after Adam's turn, the first counter is $1 \mod 4$. It is easy to see that if Eve does not change the parity of the counter back to zero with her following turn, then Adam has a winning strategy. Indeed, in this case, he will play his POSITIVITY CHECK if and only if the first counter is not $3 \mod 4$. Eve cannot make the counter 0, as she cannot even make it $0 \mod 4$. Thus, Eve has to play a move adding -1 to the first counter. The only move for that is $\langle\!\langle s_{ab}, (-1, 0), \top_{ab} \rangle\!\rangle$ which takes Eve to an emptying state. In the first case, the emptying state is $\top_{0b}$ and all the transitions from it do not modify the first counter, i.e., Eve cannot reach $(0, 0)$. In the second case, the emptying state is $\top_{+b}$, the next transition subtracts 4 from the first counter making it negative, and there are no moves that increment the counters. Again, Eve cannot reach the origin. The case where Eve makes a mistake with the second counter is proven analogously and, in fact, Adam's strategy is the same. □

Lemma 5.4. *Assume that Eve plays only correct* SIMULATING MOVES *before Adam plays the* POSITIVITY CHECK *for the first time. If Adam plays the* POSITIVITY CHECK*, then Eve has a winning strategy starting with an* EMPTYING MOVE.

Proof. Similarly as in the previous proof, if Eve does not play an EMPTYING MOVE, then Adam has a winning strategy. Now, the configuration is $[s_{ab}, (4x + 1, y)]$ after Adam's turn

and Eve plays $\langle\!\langle s_{ab}, (-1, 0), \top_{ab} \rangle\!\rangle$. From that point onward, Eve can empty the counters ensuring that the first counter is $0 \mod 4$ and that the flags match the positivity of the counters. That is, every time Adam plays his POSITIVITY CHECK, Eve plays an EMPTYING MOVE subtracting one from the first counter. Eventually, Eve will reach the configuration $[\top_{00}, (0, 0)]$ and win the game. □

Lemma 5.5. *If Adam plays only the* 0-MOVE *and Eve plays an* EMPTYING MOVE*. Adam has a winning strategy starting with the* 0-MOVE*.*

Proof. After Eve's move, the first counter is $3 \mod 4$. As in the proof of Lemma 5.3, Adam ensures that the first counter stays non-zero modulo four and wins the game. □

We are ready to prove the main theorem.

Theorem 5.1. *Let (Q, T, s_0) be a two-counter machine. There exists a two-dimensional* $\mathbb{Z}$-VASS *game (A, E, c_0) where Eve has a winning strategy if and only if (Q, T, s_0) reaches a configuration in $Q \times \{(0, 0)\}$.*

Proof. Let (A, E, c_0) be the $\mathbb{Z}^2$-VASS game constructed in this section. Assume first that (Q, T, s_0) reaches a configuration in $Q \times \{(0, 0)\}$. Now by Lemma 5.2, Eve's winning strategy is to respond with the correct SIMULATING MOVES if Adam plays the 0-MOVE, and if Adam plays the POSITIVITY CHECK, then Eve has a sequence of moves described in Lemma 5.4 that leads to the configuration $[\top_{00}, (0, 0)]$.

Assume then that (Q, T, s_0) never reaches a configuration in $Q \times \{(0, 0)\}$. We show that Eve does not have a winning strategy. If Adam plays only the 0-MOVE, then, by Lemma 5.2, Eve does not win by responding with just the correct SIMULATING MOVES. Alternatively, if at some point, she plays either an incorrect SIMULATING MOVE or an EMPTYING MOVE, then by Lemma 5.3 and Lemma 5.5, respectively, Adam has winning strategies making sure that a configuration with counter values $(0, 0)$ is not reachable. As we analysed all the possible moves of Eve, we have shown that Eve does not have a winning strategy. □

By Theorem 2.4 and Theorem 5.1, we have the following corollary regarding decidability of games on two-dimensional integer vector addition systems with states.

Corollary 5.6. *Let (A, E, c_0) be a* $\mathbb{Z}^2$-VASS *game. It is undecidable whether Eve has a winning strategy to reach $[s, t, (0, 0)]$, for any $s \in Q_E$ and $t \in Q_A$, from c_0. The problem is undecidable even when Adam is stateless and does not modify the second counter.*

5.2 $\mathbb{Z}$-VAS games in two dimensions

In this section, we prove the main result of the chapter. We prove that it is undecidable whether Eve has a winning strategy in a game on a two-dimensional integer vector addition system. We prove the claim by constructing a $\mathbb{Z}^2$-VAS game that simulates a $\mathbb{Z}^2$-VASS game. In some ways, the construction is similar to the construction of a game with states in the previous section as can be seen in similarities of Figure 5.1 and Figure 5.4. The construction of the stateless game is more complex as the information on two counters, states and state transitions has to be embedded into two-dimensional vectors.

Theorem 5.7. *Let (A_1, E_1, c_0) be a two-dimensional $\mathbb{Z}$-VASS game, where Adam is stateless and does not modify the second counter. There exists a two-dimensional $\mathbb{Z}$-VAS game $(A, E, \mathbf{x}_0)$ where Eve has a winning strategy if and only if Eve has a winning strategy in (A_1, E_1, c_0).*

Similarly to the construction of Section 5.1, the idea is that, in the $\mathbb{Z}^2$-VAS game, Eve and Adam simulate a play of the $\mathbb{Z}^2$-VASS game where Adam is stateless and does not modify the second counter. As Adam is stateless, we again simplify the notation and write a configuration of the $\mathbb{Z}^2$-VASS game as $[s, \mathbf{x}]$, where $s \in Q_E$ and $\mathbf{x} \in \mathbb{Z}^2$.

In the $\mathbb{Z}^2$-VAS game, if one of the players deviates from the play of the $\mathbb{Z}^2$-VASS game, the opponent has a winning strategy from that point onward. In Figure 5.4, we present a schema similar to Figure 5.1, depicting the possible ways two-dimensional $\mathbb{Z}$-VAS games can go. Three of the outcomes have a predetermined winner, which does not depend on the $\mathbb{Z}^2$-VASS game. In the last case, where Eve and Adam correctly simulate the $\mathbb{Z}^2$-VASS game, the winner depends on the winner of the $\mathbb{Z}^2$-VASS game, i.e., whether Eve has a winning strategy to reach $[s, (0, 0)]$, for some state s, or not.

- If Eve's move corresponds to a move in a play of the $\mathbb{Z}^2$-VASS game, that we call a REGULAR MOVE, and Adam replies with his REGULAR MOVE, then iteratively applying only this turn-based interaction, Eve has a winning strategy if and only if she has a winning strategy in the corresponding $\mathbb{Z}^2$-VASS game (Lemma 5.9).

- If Eve's move incorrectly simulates the $\mathbb{Z}^2$-VASS game, then Adam has a winning strategy from this moment on starting with a STATE-CHECK that makes Eve's target unreachable (Lemma 5.10).

- If Adam plays his STATE-CHECK following a correct REGULAR MOVE of Eve, then Eve has a winning strategy from this moment on starting with a STATE-DEFENCE MOVE allowing Eve to empty both counters and reach $(0, 0)$ (Lemma 5.11).

- Finally, if Eve plays a STATE-DEFENCE MOVE instead of a REGULAR MOVE, in that case Adam has a winning strategy starting by playing his REGULAR MOVE (Lemma 5.12).

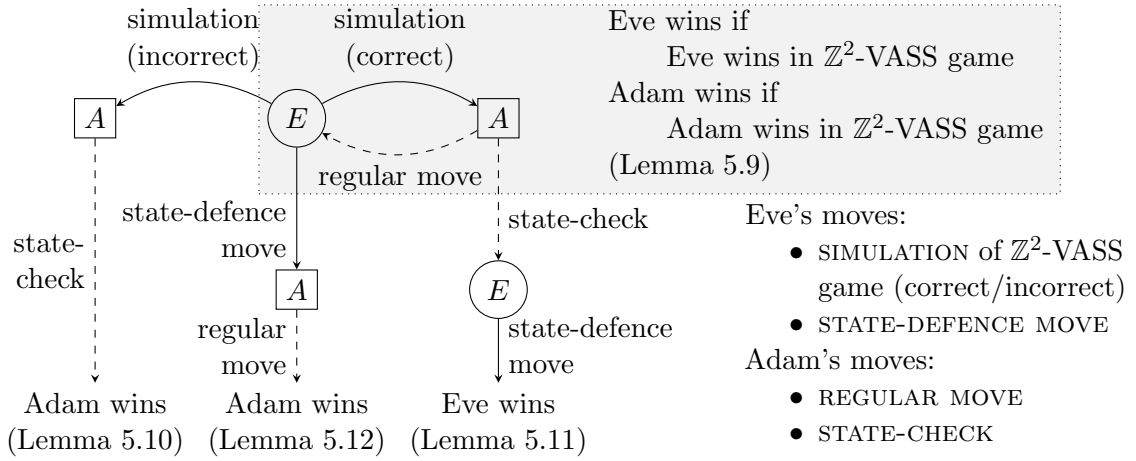


Figure 5.4: Progress of a $\mathbb{Z}^2$-VAS game.

Intuitively, we encode the states in the second counter as powers of 8 such that the coefficient of 8^i is 1 if and only if Eve's state in $\mathbb{Z}^2$-VASS game is s_i. When the state changes from s_i to s_j, $-8^i + 8^j$ is added to the second counter. We represent states as coefficients of powers of eight because we need the extra space smaller bases do not possess.

It is easy to see that simply encoding states as powers of 8 is not enough as incorrect transitions can result in a correct configuration. For example, if the configuration of the $\mathbb{Z}^2$-VASS is $[s_i, (x, y)]$ and moves corresponding to $\langle\!\langle s_j, (a, b), s_k \rangle\!\rangle$ and $\langle\!\langle s_k, (c, d), s_j \rangle\!\rangle$ are used, the resulting configuration corresponds to $[s_i, (x + a + c, y + b + d)]$. Another way to cheat is to use carries as incrementing the coefficient of 8^i eight times is indistinguishable from incrementing the coefficient of 8^{i+1} once. Both types of cheating can be countered with Adam's STATE-CHECKS.

We now show how we embed the states and state transitions into the second counter of the game. Similarly to how in the previous section we created additional space in the first counter by multiplying the moves modifying the first counter by four, we multiply the

second counter by $4 \cdot 8^n$, where $n = m + 7$ and m is the number of states, creating enough space to store all the needed information of the underlying automaton. The multiplication by $4 \cdot 8^n$ rather than just 8^n has two purposes. The first one is similar to multiplying the first counter by four in Section 5.1. Namely, certain moves will move between different intervals modulo $4 \cdot 8^n$ ensuring the correct response from the opponent. This is illustrated in Figure 5.5. The second purpose is to ensure that above described cheating with carries is not possible. A configuration of a $\mathbb{Z}^2$-VASS game is mapped to a vector in $\mathbb{Z}^2$ by $[s_i, (c_1, c_2)] \mapsto (c_1, c_2 \cdot 4 \cdot 8^n + 8^i)$.

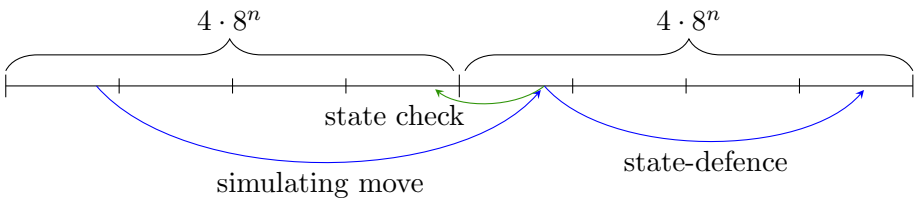


Figure 5.5: An illustration of changes in interval when simulating or state-defence moves of Eve or state check of Adam are applied.

Before presenting the detailed constructions of Eve's and Adam's moves, we note that we can assume that the $\mathbb{Z}^2$-VASS game has the information on the positivity of the counters, and players have to update the information correctly. Indeed, this was done in the previous section by using flags 0 and $+$. Recall that, because of this, the first counter is incremented and decremented by 4. By this assumption, we can denote the states of Eve by s_{ab} as before. We also assume that Eve's underlying graph is without self-loops as they would allow Eve to modify the counters without modifying coefficients of the states. Let Q be the set of states of Eve in a $\mathbb{Z}^2$-VASS game. We create an emptying gadget for Eve similar to the one constructed in the previous reduction. To avoid self-loops, there are seven emptying states, $\{\top_{ab}, \top'_{ab} \mid a, b \in \{0, +\}\} \setminus \{\top'_{00}\}$. The state $\top'_{00}$ is not needed as $\top_{00}$ will not have any moves from it. The moves in the emptying gadget are as in the emptying gadget constructed in Section 5.1 but instead of self-loops, the transitions are between primed and unprimed versions of the states, which is similar to the construction of Lemma 3.8. The emptying moves can be found in Table 5.2

We denote $\mathcal{T} = \{\top_{++}, \top'_{++}, \top_{0+}, \top'_{0+}, \top_{+0}, \top'_{+0}\}$. We think of elements of $Q \cup \mathcal{T} \cup \{\top_{00}\}$ as integers in $\{0, \dots, n-1\}$ such that $\top_{00} = 0$, $\top'_{0+} = n-6$, $\top_{0+} = n-5$, $\top'_{+0} = n-4$, $\top_{+0} = n-3$, $\top'_{++} = n-2$ and $\top_{++} = n-1$. We give names for update vectors that we

$\{\langle\!\langle \top_{++}, (-4,-1)-\alpha, t\rangle\!\rangle \mid \alpha \in A_1, t \in \{\top'_{++}, \top_{+0}, \top'_{+0}, \top_{0+}, \top'_{0+}, \top_{00}\}\}$; $\{\langle\!\langle \top'_{++}, (-4,-1)-\alpha, t\rangle\!\rangle \mid \alpha \in A_1, t \in \{\top_{++}, \top_{+0}, \top'_{+0}, \top_{0+}, \top'_{0+}, \top_{00}\}\}$;
$\{\langle\!\langle \top_{+0}, (-4,0)-\alpha, t\rangle\!\rangle \mid \alpha \in A_1, t \in \{\top'_{+0}, \top_{00}\}\}$; $\{\langle\!\langle \top'_{+0}, (-4,0)-\alpha, t\rangle\!\rangle \mid \alpha \in A_1, t \in \{\top_{+0}, \top_{00}\}\}$;
$\{\langle\!\langle \top_{0+}, (0,-1)-\alpha, t\rangle\!\rangle \mid \alpha \in A_1, t \in \{\top'_{0+}, \top_{00}\}\}$; $\{\langle\!\langle \top'_{0+}, (0,-1)-\alpha, t\rangle\!\rangle \mid \alpha \in A_1, t \in \{\top_{0+}, \top_{00}\}\}$;

Table 5.2: The modified emptying gadget of a $\mathbb{Z}^2$-VASS game.

often use:

$$\textsc{Add}(1,x) := (x,0); \qquad \textsc{Move}(j,k) := (0, -8^j + 8^k), \text{ for } 0 \le j,k \le n-1;$$
$$\textsc{Add}(2,x) := (0, 4\,x \cdot 8^n); \quad \textsc{Check}(i) := (0, -5 \cdot 8^i - 8^n), \text{ for } n-6 \le i \le n-1.$$

The initial vector $\mathbf{x}_0$ of the $\mathbb{Z}^2$-VAS game is $\textsc{Add}(1,x) + \textsc{Add}(2,y) + \textsc{Move}(\top_{00}, s)$, that is, $\mathbf{x}_0 = (x, 4\,y \cdot 8^n + 8^s - 8^0)$, where $[s,(x,y)]$ is the initial configuration in the $\mathbb{Z}^2$-VASS game.

In the next example, we illustrate how the update vectors modify the counters.

Example 5.8. Let (A_1, E_1, c_0) be a $\mathbb{Z}^2$-VASS game, where Eve has two states, $s = 1$ and $t = 2$, and the initial configuration $c_0 = [s,(1,0)]$. In Figure 5.6, we present a set of configurations in the $\mathbb{Z}^2$-VAS game obtained from the corresponding initial configuration when we apply $\textsc{Add}(1,-1)$, $\textsc{Add}(2,1)$, $\textsc{Move}(s,t)$ and $\textsc{Check}(8)$ in succession.

Now we present the moves of the players. Adam has two types of moves: REGULAR MOVES that correspond to the moves in the $\mathbb{Z}^2$-VASS game and STATE-CHECK MOVES, $\{\textsc{Check}(i) \mid i \in \mathcal{T}\}$. The moves of Eve correspond to moves in E_1 where incrementing and decrementing of the second counter is by $4 \cdot 8^n$ rather than by 1. Let $\langle\!\langle s,(x,y),t\rangle\!\rangle \in E_1$, then

$$\textsc{Add}(1,x) + \textsc{Add}(2,y) + \textsc{Move}(s,t) = (x, 4y \cdot 8^n - 8^s + 8^t) \in E.$$

We call these moves REGULAR MOVES. We also need a move for Eve to finish the simulation by removing any values corresponding to the automaton if the state is s_{00}. That is, we add moves $\{\textsc{Move}(s_{00}, \top_{00}) - \alpha \mid \alpha \in A_1\}$. The other type of moves, STATE-DEFENCE MOVES, are used to empty the counters. As in the previous construction, Eve will be able to cancel every move of Adam and decrement the counters at the same time. The full list of moves can be found in Table 5.3.

$$
\begin{array}{c}
\overbrace{(1,0\cdot 4\cdot 8^9}^{\mathbb{Z}^2\text{-VASS game counters}} + \overbrace{0\cdot 8^8+0\cdot 8^7+0\cdot 8^6+0\cdot 8^5+0\cdot 8^4+0\cdot 8^3}^{\mathcal{T}} + \overbrace{0\cdot 8^2+1\cdot 8^1}^{\text{states of } \mathbb{Z}^2\text{-VASS game}} \overbrace{-1\cdot 8^0)}^{\top_{00}} \\
\downarrow \textsc{Add}(1,-1) \\
(0,0\cdot 4\cdot 8^9+0\cdot 8^8+0\cdot 8^7+0\cdot 8^6+0\cdot 8^5+0\cdot 8^4+0\cdot 8^3+0\cdot 8^2+1\cdot 8^1-1\cdot 8^0) \\
\downarrow \textsc{Add}(2,1) \\
(0,4\cdot 8^9+0\cdot 8^8+0\cdot 8^7+0\cdot 8^6+0\cdot 8^5+0\cdot 8^4+0\cdot 8^3+0\cdot 8^2+1\cdot 8^1-1\cdot 8^0) \\
\downarrow \textsc{Move}(s,t) \\
(0,4\cdot 8^9+0\cdot 8^8+0\cdot 8^7+0\cdot 8^6+0\cdot 8^5+0\cdot 8^4+0\cdot 8^3+1\cdot 8^2+0\cdot 8^1-1\cdot 8^0) \\
\downarrow \textsc{Check}(8) \\
(0,3\cdot 8^9-5\cdot 8^8+0\cdot 8^7+0\cdot 8^6+0\cdot 8^5+0\cdot 8^4+0\cdot 8^3+1\cdot 8^2+0\cdot 8^1-1\cdot 8^0).
\end{array}
$$

Figure 5.6: Applying vectors $\textsc{Add}(1,-1)$, $\textsc{Add}(2,1)$, $\textsc{Move}(s,t)$ and $\textsc{Check}(8)$ in succession to a vector corresponding to configuration $[s,(1,0)]$ of a $\mathbb{Z}^2$-VASS game.

Finally, we define moves connecting the simulating partition of Eve's automaton with the emptying partition. For each state $s_{ab} \in Q$, where a, b are not both zero, Eve has a move $\{\textsc{Move}(s_{ab},k) - \textsc{Check}(i) \mid (a,b) \in \{0,+\}^2 \setminus \{(0,0)\}, k \in \{\top_{ab}, \top'_{ab}\}, k \neq i, i \in \mathcal{T}\}$. For s_{00}, Eve has a move $\{\textsc{Move}(s_{00},\top_{00}) - \textsc{Check}(i) \mid i \in \mathcal{T}\}$.

Next, we prove which player has a winning strategy in the scenarios presented previously.

Lemma 5.9. *If both players only play* REGULAR MOVES *and Eve plays only correct* REGULAR MOVES*, then Eve has a winning strategy if and only if she has a winning strategy in the* $\mathbb{Z}^2$-VASS *game.*

Proof. It easy to see that REGULAR MOVES of the players simulate the $\mathbb{Z}^2$-VASS game and that Eve has a winning strategy to reach a configuration $[s_{00},(0,0)]$ of the $\mathbb{Z}^2$-VASS game if and only if she has a winning strategy to reach the vector $(0,0\cdot 4\cdot 8^n+8^{s_{00}}-8^{\top_{00}})$ in the $\mathbb{Z}^2$-VAS game after which Eve wins by playing $\textsc{Move}(s_{00},\top_{00}) - \alpha$, where α is the REGULAR MOVE played by Adam. □

Lemma 5.10. *If Eve plays an incorrect move, i.e., after her turn the coefficient of some* 8^s *is* -1 *or the coefficient of* $8^{\top_{00}}$ *is zero, then Adam has a winning strategy starting with a* STATE-CHECK.

Adam's move	Eve's move
$\alpha \in A_1$	$\{\textsc{Add}(1,-4) + \textsc{Add}(2,-1) + \textsc{Move}(j,k) - \alpha \mid j \in \{\top_{++}, \top'_{++}\}, k \in \mathcal{T}, j \neq k\}$
	$\{\textsc{Add}(1,-4) + \textsc{Move}(j,k) - \alpha \mid j,k \in \{\top_{+0}, \top'_{+0}\}, j \neq k\}$
	$\{\textsc{Add}(2,-1) + \textsc{Move}(j,k) - \alpha \mid j,k \in \{\top_{0+}, \top'_{0+}\}, j \neq k\}$
	$\{\textsc{Add}(1,-4) + \textsc{Add}(2,-1) + \textsc{Move}(j,0) - \alpha \mid j \in \{\top_{++}, \top'_{++}\}\}$
	$\{\textsc{Add}(1,-4) + \textsc{Move}(j,0) - \alpha \mid j \in \{\top_{+0}, \top'_{+0}\}\}$
	$\{\textsc{Add}(2,-1) + \textsc{Move}(j,0) - \alpha \mid j \in \{\top_{0+}, \top'_{0+}\}\}$
$\textsc{Check}(i)$	$\{\textsc{Add}((1,-4e_1) + \textsc{Add}(2,-e_2) - \textsc{Check}(i) \mid e_1, e_2 \in \{0,1\}\}$
	$\{\textsc{Add}(1,-4) + \textsc{Add}(2,-1) + \textsc{Move}(j,k) - \textsc{Check}(i) \mid i, j \neq k,$ $j \in \{\top_{++}, \top'_{++}\}, k \in \mathcal{T}\}$
	$\{\textsc{Add}(1,-4) + \textsc{Add}(2,-1) + \textsc{Move}(j,0) - \textsc{Check}(i) \mid j \in \{\top_{++}, \top'_{++}\}\}$
	$\{\textsc{Add}(1,-4) + \textsc{Move}(j,0) - \textsc{Check}(i) \mid j \in \{\top_{+0}, \top'_{+0}\}\}$
	$\{\textsc{Add}(2,-1) + \textsc{Move}(j,0) - \textsc{Check}(i) \mid j \in \{\top_{0+}, \top'_{0+}\}\}$

Table 5.3: STATE-DEFENCE MOVES of Eve.

Proof. First, we prove that Eve loses if a coefficient corresponding to a state of the $\mathbb{Z}^2$-VASS game is negative after one of her turns. A coefficient corresponding to a state of the $\mathbb{Z}^2$-VASS game can only be increased, namely incremented, by Eve's REGULAR MOVES. Hence, if one of the coefficients becomes negative, then Adam wins by playing a STATE-CHECK move. The reasoning is now similar to the usage of the POSITIVITY CHECK in Lemma 5.3. We consider the second counter modulo $4 \cdot 8^n$. Before Adam's STATE-CHECK, the configuration is in $[0, 8^n) \mod 4 \cdot 8^n$ and after the check in $[3 \cdot 8^n, 4 \cdot 8^n) \mod 4 \cdot 8^n$. If Eve does not play a STATE-DEFENCE MOVE (a move containing a $\textsc{Check}(i)$), then Adam has a winning strategy by playing a STATE-CHECK if the second counter is not in $[3 \cdot 8^n, 4 \cdot 8^n) \mod 4 \cdot 8^n$ and a REGULAR MOVE otherwise (recall that Adam's REGULAR MOVES do not modify the second counter). Thus, Eve has to play a STATE-DEFENCE MOVE which does not make the negative coefficient non-negative. Now at least one of the coefficients in $\mathcal{T}$ is non-zero, say i. Adam will play $\textsc{Check}(i)$ forcing Eve to play a move containing $-\textsc{Check}(i)$ which will make another coefficient in $\mathcal{T}$ non-zero (or keep the same coefficient non-zero). As long as Adam keeps playing the correct STATE-CHECK, Eve cannot make all the coefficients zero and thus cannot win.

The second case where a coefficient of some state in $\mathcal{T}$ is negative has been proven above. For the final case, where the coefficient of $8^{\top_{00}}$ is zero, we consider the next move of Eve. During her next turn, Eve has to play a move containing $\textsc{Move}(s,t)$ making the coefficient of 8^s negative, which has been covered previously. □

Lemma 5.11. *Assume that Eve plays only correct* REGULAR MOVES *before Adam plays a* STATE-CHECK *for the first time. If Adam plays a* STATE-CHECK*, then Eve has a winning strategy starting with a* STATE-DEFENCE MOVE*.*

Proof. Similarly as in the previous proof, if Eve does not play a STATE-DEFENCE MOVE, then Adam has a winning strategy. Now, Eve plays the STATE-DEFENCE MOVE $\textsc{Move}(s_{ab}, k) - \textsc{Check}(i)$, where s_{ab} is the non-zero coefficient, $\textsc{Check}(i)$ is the STATE-DEFENCE MOVE Adam played and $k \in \{\top_{ab}, \top'_{ab}\}$, $k \neq i$. From that point onward, Eve can empty the counters as she has emptying moves with an opposite move of Adam. Eventually, Eve will reach the configuration $(0, 0)$ and win the game. □

Lemma 5.12. *If Adam plays only* REGULAR MOVES *and Eve plays a* STATE-DEFENCE MOVE*, then Adam has a winning strategy starting with a* REGULAR MOVE*.*

Proof. There are two cases to consider. Either the STATE-DEFENCE MOVE contains $\textsc{Check}(i)$ or not. In the first case, since all such STATE-DEFENCE MOVES subtract -8^n from the second counter, after Eve's move, the counter is in $[8^n, 2 \cdot 8^n) \mod 4 \cdot 8^n$. As in the proof of Lemma 5.10, Adam ensures that the second counter does not return to the interval $[0, 8^n) \mod 4 \cdot 8^n$. In the second case, the STATE-DEFENCE MOVE contains $\textsc{Move}(j, k)$ for some $j \in \mathcal{T}$, and then by Lemma 5.10, Adam has a winning strategy. □

We are ready to prove the main theorem of the chapter.

Theorem 5.7. *Let (A_1, E_1, c_0) be a two-dimensional* $\mathbb{Z}$-VASS *game where Adam is stateless and does not modify the second counter. There exists a two-dimensional* $\mathbb{Z}$-VAS *game $(A, E, \mathbf{x}_0)$ where Eve has a winning strategy if and only if Eve has a winning strategy in (A_1, E_1, c_0).*

Proof. Let $(A, E, \mathbf{x}_0)$ be the $\mathbb{Z}^2$-VAS game constructed in this section. Assume first that Eve has a winning strategy in (A_1, E_1, c_0). Now, Eve's winning strategy in the two-dimensional $\mathbb{Z}^2$-VAS game is to follow the strategy of (A_1, E_1, c_0) as long as Adam plays REGULAR MOVES which is a winning strategy by Lemma 5.9. If Adam plays a STATE-CHECK, then Eve responds according to the winning strategy of Lemma 5.11.

Assume then, towards a contradiction, that Adam has a winning strategy in (A_1, E_1, c_0) and Eve has a winning strategy in $(A, E, \mathbf{x}_0)$. If Adam plays only REGULAR MOVES, then by Lemma 5.9, Eve does not win by playing just the correct SIMULATING MOVES. That is, Eve has to, at some point, either play an incorrect SIMULATION MOVE or play a STATE-DEFENCE

MOVE. By Lemma 5.10 and Lemma 5.12, Adam has winning strategies for both cases. As we analysed all the possible moves of Eve, we have shown that Eve does not have a winning strategy. □

Corollary 5.13. *Let $(A, E, \mathbf{x}_0)$ be a two-dimensional $\mathbb{Z}^2$-VAS game. It is undecidable whether Eve has a winning strategy to reach $(0, 0)$ from $\mathbf{x}_0$.*

Corollary 5.13 follows from Corollary 5.6 and Theorem 5.7.

5.3 VAS games in two dimensions

In this section, we consider VAS games, i.e., where the arena is limited to the positive quadrant $\mathbb{Z}^{+2}$. We prove that it is undecidable whether Eve has a winning strategy in two-dimensional VAS games. Recall that the problem is undecidable for two-dimensional VASS games, where the game is played on a vector addition system with states [1, 29]. Our proof relies on the construction of the previous section.

In the $\mathbb{Z}^2$-VAS game constructed in the previous section, the initial vector is in $\mathbb{Z}^{+2}$ and both players have moves containing $\textsc{Check}(i) := (0, -5 \cdot 8^i - 8^n)$, for $n - 6 \leq i \leq n - 1$ for some $n \in \mathbb{Z}^+$. Adam's moves contain $\textsc{Check}(i)$ and Eve's moves contain $-\textsc{Check}(i)$. Other than these moves, all other moves of Adam have only non-negative components. By considering $\textsc{Check}'(i) := (0, 5 \cdot 8^i + 8^n)$, we have a variant of the game where Adam's moves have only non-negative components. In other words, VAS semantics do not affect Adam's behaviour.

Theorem 5.14. *Let $(A, E, \mathbf{x}_0)$ be a two-dimensional* VAS *game. It is undecidable whether Eve has a winning strategy to reach $(0, 0)$ from $\mathbf{x}_0$.*

Proof. Consider the two-dimensional $\mathbb{Z}^2$-VAS game $(A_1, E_1, \mathbf{x}_0)$ of Corollary 5.13. We modify the sets of moves by changing moves containing $\textsc{Check}(i)$ into $\textsc{Check}'(i)$. After this, as all moves of Adam are positive, only Eve can reach a deadlock. Other than deadlocks, the proof goes as in Section 5.2. □

5.4 Concluding remarks and open problems

In this chapter, we proved the main result of the thesis. We showed that it is undecidable which player has a winning strategy in $\mathbb{Z}^2$-VAS game. Additionally, we proved that the result holds under VAS semantics as well.

Korec showed in [99] that there exists a universal Minsky machine with 32 instructions, in the sense that the universal machine can simulate any other Minsky machine. The natural question of a universal game arises: Is it possible to construct a fixed $\mathbb{Z}^2$-VAS game simulating a universal 2CM? This game would have fixed moves and only the initial vector would affect the result.

Consider the universal Minsky machine with 32 instructions. We can construct a $\mathbb{Z}^2$-VAS game from it and count the number of moves. Thus, it is undecidable whether Eve has a winning strategy in a two-dimensional $\mathbb{Z}^2$-VAS game where Eve has at least 2419 moves and Adam has eight moves. The optimal bounds on number of players' moves between decidability and undecidability remain an open problem.

Chapter 6

Controllability in $\mathbb{Z}^d$-VASS games

The Attacker-Defender games we have considered so far have been either undecidable or lie in high complexity classes such as EXPTIME or EXPSPACE. In this chapter, we investigate modifications to $\mathbb{Z}^d$-VASS games that could yield lower complexity bounds whilst remaining nontrivial.

We observe that the gadgets present in Chapter 4 and Chapter 5 that ensure a faithful simulation, rely on an infinite punishing behaviour by Adam. For example, in Section 5.1, if Eve enters the emptying gadget prior to Adam playing $(1, 0)$, then Adam has a simple winning strategy, where he has to ensure that the first counter is not 3 mod 4. But if we limit how many times Adam can play, then Eve can exhaust Adam's moves and then reach $(0, 0)$ with an unfaithful simulation. Motivated by this behaviour, in this chapter, we consider two variants of $\mathbb{Z}^d$-VASS games where players' resources are limited.

- k-control $\mathbb{Z}^d$-VASS games with a safety objective, where Adam can apply his moves only k times during any play and has to skip otherwise.
- k-control $\mathbb{Z}^d$-VASS games with a reachability objective, where Eve can apply her moves only k times during any play and has to skip otherwise.

The results are highlighted in Table 6.1. Note, that k is fixed and is not part of the input.

Example 6.1. Let (A, E, x_0) be a $\mathbb{Z}$-VAS game, where $A = \{-2\}$, $E = \{1\}$ and $x_0 < 0$. In the $\mathbb{Z}$-VAS game, Adam has a winning strategy as, after the players play their moves, the resulting configuration $x = x_0 - 2 + 1$ is less than the initial configuration x_0. If we consider it as a k-control $\mathbb{Z}$-VAS game with a safety objective for any $k \in \mathbb{Z}^+$, then Eve

Game	$k = 0$	k	$k = \infty$	
			$d = 1$	$d \geq 2$
$\mathbb{Z}^d$-VAS games (safety)	NP-c [77]	NP-c (Corollary 6.8)	EXPTIME-c [9]	undecid. (Theorem 5.14)
$\mathbb{Z}^d$-VAS games (reachability)	PTIME	PTIME (Theorem 6.12)		
$\mathbb{Z}^d$-VASS games (safety)	NP-c [77,137]	NP-c (Corollary 6.7)	EXPSPACE-c (Theorem 4.4)	undecid. [137]
$\mathbb{Z}^d$-VASS games (reachability)	PTIME	NP (Theorem 6.13)		

Table 6.1: Summary of results on k-control games. Note that the games with safety and reachability objectives are the same for $k = \infty$.

has a winning strategy. Indeed, with at most $2k + |x_0|$ moves, Eve can reach the origin as Adam can apply his -2 move at most k times. Two plays are depicted in Figure 6.1.

Figure 6.1: A play of the $\mathbb{Z}$-VAS game of Example 6.1 (left) and a play in the 1-control $\mathbb{Z}$-VAS game, where Adam can play his move only once (right).

The main result of the chapter is that checking whether Adam has a winning strategy to ensure safety of a given configuration is NP-complete. The main observation is that, if Adam has a winning strategy in a d-dimensional k-control $\mathbb{Z}^d$-VASS game, then he has a winning strategy where his moves are played "close" to the target configuration. The intuition is clear, if the configuration is "far" from the target, then Adam does not have to use one of his moves to influence the play. On the other hand, if the configuration is "close" to the target, then Adam has to play a move in order to avoid the target. The area "close" to the target is polynomial in size, and to check whether Eve can reach it is in NP.

The second variant of the k-control $\mathbb{Z}^d$-VASS game, where Eve has limited number of moves, relies on different observations. We notice that if Eve skips her move, then Adam has a winning strategy as he can always choose a vector such that the zero vector is not reachable. On the other hand, as Eve can play only k times, eventually, she will have to only skip. That is, only a finite duration of the game needs to be considered.

Throughout the chapter, we will denote $|A| = \ell$ and $|E| = m$.

6.1 Safety of k-control $\mathbb{Z}^d$-VASS games

In this section, we consider k-control $\mathbb{Z}^d$-VASS games with a safety objective. As Adam can skip his turn, we have a lower bound from the reachability problem for $\mathbb{Z}^d$-VASS. In [77], it was proved that the reachability problem is NP-complete. Thus, k-control $\mathbb{Z}^d$-VASS games are NP-hard.

Let us then study 1-control $\mathbb{Z}^d$-VASS games to illustrate the idea of the proof better. We denote $\overline{E} = \{(s, -\mathbf{e}) \mid \langle\!\langle s, \mathbf{e}, s' \rangle\!\rangle \in E\}$, where E is the move set of Eve. The set $\overline{E}$ consists of pairs in $Q_E \times \mathbb{Z}^d$ from which she can win with one move. In the next lemma, we show that if Adam has a winning strategy, he also has a winning strategy, where Adam plays his move only if the configuration is $[s, \dot{t}, \mathbf{x}]$ for $(s, \mathbf{x}) \in \overline{E}$. In the proof, we consider a path reaching $(s, \mathbf{x}) \in \overline{E}$ using only moves of Eve. Because Adam has a winning strategy, then in the game following this path, he plays a move $\langle\!\langle t, \mathbf{a}, t' \rangle\!\rangle$ making $\mathbf{0}$ unreachable for Eve. If Adam plays $\langle\!\langle t, \mathbf{a}, t' \rangle\!\rangle$ in $[s, \dot{t}, \mathbf{x}]$, then Eve cannot reach the origin from the resulting configuration $[\dot{s}, t', \mathbf{x} + \mathbf{a}]$. The idea is depicted in Figure 6.2.

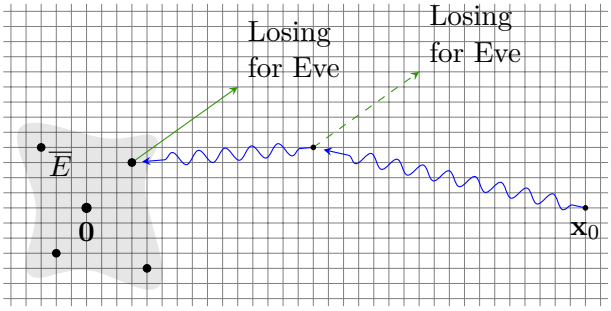


Figure 6.2: An illustration of Lemma 6.2 in a $\mathbb{Z}^2$-VAS game.

Lemma 6.2. *Let σ be a winning strategy of Adam in a 1-control $\mathbb{Z}^d$-VASS game with a safety objective $(A, E, [s_0, t_0, \mathbf{x}_0])$. Then also σ' is a winning strategy of Adam, where*

$$\sigma'([s,t,\mathbf{x}]) = \begin{cases} \textsc{Skip} & \text{if } (s,\mathbf{x}) \notin \overline{E} \\ \sigma([s',t,\mathbf{y}]) & \text{if } (s,\mathbf{x}) \in \overline{E}, \text{ for some } [s',t,\mathbf{y}]. \end{cases}$$

Proof. Let σ be a winning strategy of Adam. Consider $(s, \mathbf{x}) \in \overline{E}$ that is reachable from $[s_0, t_0, \mathbf{x}_0]$ using Eve's moves only. Let p be this path. Consider then a play in which Eve

follows p. That is,

$$p = [s_0, t_0, \mathbf{x}_0][s_{i_j}, t_0, \mathbf{x}_1] \cdots [s', t_0, \mathbf{y}] \cdots [s, t_0, \mathbf{x}].$$

Since σ is a winning strategy, there is a configuration $[s', t_0, \mathbf{y}]$ for which the winning strategy $\sigma([s', t_0, \mathbf{y}]) = \langle\!\langle t_0, \mathbf{a}, t \rangle\!\rangle$, for some $\langle\!\langle t_0, \mathbf{a}, t \rangle\!\rangle \in A$, making $\mathbf{0}$ unreachable using Eve's moves. If $(s', \mathbf{y}) = (s, \mathbf{x})$, then we are done. If $(s', \mathbf{y}) \neq (s, \mathbf{x})$, then consider a strategy of Adam, where he skips unless $(s, \mathbf{x})$ is reached and then plays $\langle\!\langle t_0, \mathbf{a}, t \rangle\!\rangle$. This play is winning for Adam. Indeed, assume that on the contrary Eve can reach $\mathbf{0}$ from $[s, t, \mathbf{x} + \mathbf{a}]$ by a path p'. Then Eve can also reach $\mathbf{0}$ from $[s', t, \mathbf{y} + \mathbf{a}]$ by playing the same moves as in the suffix of p followed by p', which is not possible as σ is a winning strategy.

By repeating this procedure for every $(s, \mathbf{x}) \in \overline{E}$, we find a move of Adam that, according to σ, Adam needs to play in $[s, t_0, \mathbf{x}]$. We construct σ' such that $\sigma'([s, t_0, \mathbf{x}]) = \textsc{Skip}$ if $(s, \mathbf{x}) \notin \overline{E}$ and $\sigma'([s, t_0, \mathbf{x}]) = \sigma([s', t_0, \mathbf{y}])$ if $(s, \mathbf{x}) \in \overline{E}$. By the previous argument, σ' is a winning strategy of Adam. □

Now, by the previous lemma, it is enough to only consider strategies where Adam always skips or where an element of $\overline{E}$ is reached after which Adam plays his move.

Let us consider the game with $k = 0$ in more details, that is, reachability in $\mathbb{Z}^d$-VASS. In [77], it was shown that deciding whether $(q_f, \mathbf{0})$ is reachable from $(q_0, \mathbf{x}_0)$ is decidable in NP. We call this algorithm $\texttt{ZVASS}(E, (q_0, \mathbf{x}_0), (q_f, \mathbf{0}))$. In $\mathbb{Z}^d$-VASS games, the target is just the counter value $\mathbf{0}$. This does not increase the complexity as Eve can guess the state q_f, where the counter becomes $\mathbf{0}$, and call $\texttt{ZVASS}(E, (q_0, \mathbf{x}_0), (q_f, \mathbf{0}))$.

input : A 1-control $\mathbb{Z}^d$-VASS game $(A, E, [s_0, t_0, \mathbf{x}_0])$.
output : The player that has a winning strategy in the safety game.

```
1 if ZVASS(E, (s_0, x_0), (s, 0)) = Adam forall s ∈ Q_E then
2 |   return Adam;
3 else
4 |   foreach element (s, e) ∈ Ē do
5 |   |   if ZVASS(E, (s_0, x_0), (s, e)) = Eve and forall ⟨⟨t_0, a, t⟩⟩ ∈ A,
  |   |     ZVASS(E, (s, e + a), (s', 0)) = Eve then return Eve;
6 |   end
7 |   return Adam;
8 end
```

Algorithm 6.1: Solving a 1-control $\mathbb{Z}^d$-VASS game with a safety objective.

Theorem 6.3. *It is decidable in* NP*, which player has a winning strategy in a 1-control* $\mathbb{Z}^d$*-*VASS *game with a safety objective for Adam.*

Proof. We claim that Algorithm 6.1 solves the problem in non-deterministic polynomial time.

First, we prove the correctness of the algorithm. Assume that Adam has a winning strategy. There are two types of strategies, one where Adam only skips and another where he plays one of his moves. In the first case, Adam has a winning strategy if and only if Eve cannot reach $\mathbf{0}$ from the initial configuration $[s_0, t_0, \mathbf{x}_0]$. That is, $\mathbf{0}$ is not reachable in the underlying $\mathbb{Z}^d$-VASS. This is verified on line 1. In the second case, by Lemma 6.2, it is enough to consider values in $\overline{E}$. As Adam has a winning strategy, for any value $(s, \mathbf{e}) \in \overline{E}$, that is reachable using only the moves of Eve, Adam has a move $\langle\!\langle t, \mathbf{a}, t' \rangle\!\rangle$ such that the origin is not reachable from $(s, \mathbf{e} + \mathbf{a})$. This is verified in the loop on line 4. It both cases the correct output is produced.

It is clear that the algorithm terminates. In total, there are at most $m\ell + m$ calls to solve the reachability in $\mathbb{Z}^d$-VASS. The reachability in $\mathbb{Z}^d$-VASS can be solved in non-deterministic polynomial time, and thus Algorithm 6.1 is in NP. □

We are ready to consider the general case of k-control $\mathbb{Z}^d$-VASS game $(A, E, [s_0, t_0, \mathbf{x}_0])$. Recall that k is fixed and is not part of the input. Lemma 6.2 generalises in a natural way. Denote by $\overline{E^k}$ the set of all pairs in $Q_E \times \mathbb{Z}^d$ from which the shortest path to reach an element of $Q_E \times \{\mathbf{0}\}$ is of length k. That is,

$$\overline{E^k} = \Big\{(s, \mathbf{x}) \mid (s, \mathbf{x}) \to^k (s', \mathbf{0}) \text{ for some } s' \in Q_E \text{ and } (s, \mathbf{x}) \not\to^{k'} (s'', \mathbf{0}) \text{ for any } s'' \in Q_E, k' < k\Big\}$$

The size of $\overline{E^k}$ is at most $\binom{m+k-1}{k} = \binom{m+k-1}{k}$, which is polynomial in the size of the input. The relationship between $\overline{E^2}$ and $\overline{E}$ of a $\mathbb{Z}^2$-VAS game is illustrated in Figure 6.3.

Lemma 6.4. *Let* σ *be a winning strategy of Adam in a* k*-control* $\mathbb{Z}^d$*-*VASS *game with a safety objective* $(A, E, [s_0, t_0, \mathbf{x}_0])$*. Then also* σ' *is a winning strategy of Adam, where*

$$\sigma'([s, t, \mathbf{x}]) = \begin{cases} \textsc{Skip} & \text{if } (s, \mathbf{x}) \notin \overline{E} \cup \overline{E^2} \cup \ldots \cup \overline{E^k} \\ \sigma([s', t', \mathbf{y}]) & \text{if } (s, \mathbf{x}) \in \overline{E} \cup \overline{E^2} \cup \ldots \cup \overline{E^k} \text{ for some } [s', t', \mathbf{y}]. \end{cases}$$

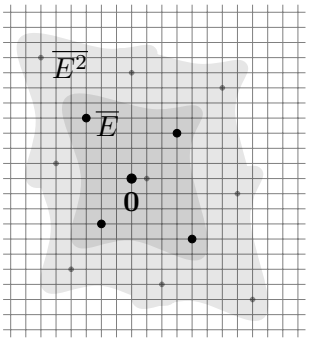


Figure 6.3: An illustration of $\overline{E^2}$ and $\overline{E}$ in a $\mathbb{Z}^2$-VAS game.

Proof. Let σ be a winning strategy of Adam. Consider $(s, \mathbf{x}) \in \overline{E^k}$ that is reachable from $[s_0, t_0, \mathbf{x}_0]$ using Eve's moves only. Because σ is a winning strategy, there are $k' \leq k$ moves of Adam application of which leads to Eve not being able to reach $Q_E \times \{\mathbf{0}\}$. Using the same reasoning as in Lemma 6.2, it is clear that Adam can play these sequence of moves starting from $\overline{E^{k'}}$ and still win.

By repeating the procedure for every element of $\overline{E} \cup \overline{E^2} \cup \ldots \cup \overline{E^k}$, we obtain a winning strategy σ', where Adam skips outside $\overline{E} \cup \overline{E^2} \cup \ldots \cup \overline{E^k}$.

□

We illustrate the need of Lemma 6.4, when $k > 1$ in the next example.

Example 6.5. Consider a k-control $\mathbb{Z}^2$-VAS game $(A, E, \mathbf{x}_0)$, where Adam's move set is $A = \{(1, 1)\}$, Eve's move set is $E = \{(-1, 0), (0, 1), (2, 1)\}$ and $\mathbf{x}_0 = (-2n, -n-1)$, for some $n > 1$, is the initial configuration. If $k = 1$, then Eve has a winning strategy. Indeed, if Adam skips and the current configuration is not $(0, -1)$, Eve plays $(2, 1)$. Then, at $(0, -1)$, Eve plays $(0, 1)$ and wins the play. Consider then a play where Adam plays his move. After his move, the configuration is $(-2n', -n'-1)+(1, 1) = (-2n'+1, -n')$ for some $0 \leq n' \leq n$. After this Eve plays $(-1, 0)$, followed by $(2, 1)$ n' times and wins the play.

In contrast, if $k > 1$, then Adam has a winning strategy. His strategy is to play $(1, 1)$ if the configuration is $(-2n', -n'-1)$ or $(-2n', -n')$ for some $n' > 0$. Note, that Adam has to play for the first time before the configuration is $(0, -1)$, as in that case Eve has a winning strategy. Two plays are depicted in Figure 6.4.

This example illustrates that this approach of $k = 1$ is not sufficient for larger k. Namely, Adam might not have a winning strategy if he only moves when the configuration is in $\overline{E}$.

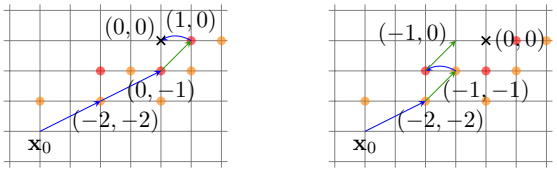


Figure 6.4: Plays of two k-control $\mathbb{Z}^d$-VAS games of Example 6.5, with $k = 1$ (left) and $k > 1$ (right). Red points are elements of $\overline{E}$ and orange are elements of $\overline{E^2}$.

Now, we are ready to prove the main result of the section.

Theorem 6.6. *It is decidable in* NP*, which player has a winning strategy in a k-control $\mathbb{Z}^d$-VASS game with a safety objective for Adam.*

Proof. We argue that Algorithm 6.2 solves the game and is in NP.

We prove the correctness of the algorithm by induction on k. If $k = 0$, then the algorithm produces the correct output as in this case Adam cannot influence the play and Eve has a winning strategy if and only if $\mathbf{0}$ is reachable in the underlying one-player $\mathbb{Z}^d$-VASS. This is checked on lines 2 and 5. Assume that the algorithm is correct for $0 \leq k' < k$. Consider then $k' + 1$. By Lemma 6.4, it is enough to consider plays, where Adam skips outside of $\bigcup_{i=1}^{k'+1} \overline{E^i}$. Assume first that Eve has a winning strategy. That is, there exists a reachable element $(s, \mathbf{e}) \in \times\overline{E^{k'+1}}$ from which Eve has a winning strategy in the k'-control $\mathbb{Z}^d$-VASS game with the initial configuration $[s, t, \mathbf{e}+\mathbf{a}]$ for any $\langle\!\langle t', \mathbf{a}, t \rangle\!\rangle$. By our induction hypothesis, the algorithm returns the correct winner of k'-control $\mathbb{Z}^d$-VASS game and thus also returns the correct result for $k' + 1$ as well. The case where Adam has a winning strategy is proved analogously.

It is clear that Algorithm 6.2 terminates. If Eve has a winning strategy, the algorithm `ControlZVASS`$(A, E, [s_0, t_0, \mathbf{x}_0], k)$ will solve at most $\binom{m+k-1}{k} + m$ instances of reachability problem of the underlying $\mathbb{Z}^d$-VASS, and make at most $\ell\binom{m+k-1}{k}$ recursive calls for `ControlZVASS`$(A, E, [s, t, \mathbf{x}], k - 1)$. The complexity of Algorithm 6.2 is $O(\ell^k m^{k^2})$, where ℓ^k and m^{k^2} are polynomial in the size of the input, together with the complexity of solving the reachability problem in the underlying $\mathbb{Z}^d$-VASS, which is in NP. □

Corollary 6.7. *Deciding which player has a winning strategy in a k-control $\mathbb{Z}^d$-VASS games with a safety objective is* NP*-complete.*

Proof. The upper bound follows from Theorem 6.6 and the lower bound from the observation in the beginning of the section by considering 0-control $\mathbb{Z}^d$-VASS games. □

```
1  function ControlZVASS(A, E, [s_0, t_0, x_0], k)
       input  : A k-control Z^d-VASS game (A, E, [s_0, t_0, x_0]).
       output : The player that has a winning strategy in the safety game.
2      if ZVASS(E, (s_0, x_0), (s, 0)) = Adam forall s ∈ Q_E then
3          return Adam;
4      else
5          if k = 0 then
6              return Eve
7          else
8              foreach element (s, e) ∈ overline(E^k) do
9                  if ZVASS(E, (s_0, x_0), (s, e)) = Eve and forall ⟪t_0, a, t⟫ ∈ A,
                     ControlZVASS(A, E, [s, t, e + a], k − 1) = Eve then return Eve;
10             end
11             return Adam;
12         end
13     end
14 end
```

Algorithm 6.2: Solving a k-control $\mathbb{Z}^d$-VASS games with a safety objective, where k is fixed.

We conclude the section with considerations of safety in k-control $\mathbb{Z}^d$-VAS games. It is clear that the upper bound from Theorem 6.6 holds for stateless games as well. The lower bound also holds as it has been proven in [77] that the reachability for $\mathbb{Z}^d$-VAS is NP-complete.

Corollary 6.8. *Deciding which player has a winning strategy in a k-control $\mathbb{Z}^d$-VAS games with a safety objective for Adam is NP-complete.*

Note, that it is possible to extract winning strategies from Algorithms 6.1 and 6.2.

6.2 Safety of k-control $\mathbb{Z}^d$-VASS games with the target defined by a hyperplane

In this section, we consider generalisation of the objective of k-control $\mathbb{Z}^d$-VASS games with a safety objective. We extend the target from a single point in $\mathbb{Z}^d$, first to a line, and then to a hyperplane, and prove that the complexity does not increase. This extension is similar to [47, 48], where the authors studied extensions of the orbit problem. Recall,

that in the orbit problem, we are given a single rational matrix and a vector and iterate the vector with the matrix. The question considered in [47] was whether a polyhedron can be reached by this iteration. The authors proved that the problem is in PSPACE, for arbitrary sized matrices and polyhedra of dimension at most two, or when both the matrix and the polyhedron is of dimension three. On the other hand, in [48], where the target is a general vector space, the problem is in PTIME if the vector space is of dimension one and in $\mathsf{NP}^{\mathsf{RP}}$ if the target space has dimension two or three. Note that both problems are over rational numbers, while we restrict ourselves to integers. Also, our model has simpler transformations, as the moves are vector additions rather than linear transformations. On the other hand, we are not restricted to a single move and, moreover, we are interested in a game scenario rather than single-player iteration.

In the first variant we study, Eve's objective is to reach a line L rather than the origin. By considering a line L with irrational slope, we can see that lower bounds follow from k-control $\mathbb{Z}^d$-VASS games with the origin as the objective. On the other hand, a line with rational slope can be added into the algorithms as an additional equation that needs to be satisfied. Indeed, for example, Algorithm 6.2 can be modified by replacing $\texttt{ZVASS}(E, (q_0, \mathbf{x}_0), (q, \mathbf{0}))$ on the second line by $\texttt{ZVASS}(E, (q_0, \mathbf{x}_0), (q, \mathbf{x}))$ and checking whether $\mathbf{x} \in L$. Clearly, the complexity does not change as a linear equation can be solved in polynomial time.

Theorem 6.9. *Let L be a line in $\mathbb{R}^d$. Deciding which player has a winning strategy in a k-control $\mathbb{Z}^d$-VASS game with safety objectives, where the targets are integer points of L, is* NP*-complete.*

Similarly, if the target is given by a system of linear equations with equalities and inequalities, then the game can be solved in non-deterministic polynomial time. A play of a k-control $\mathbb{Z}^d$-VAS game, where the objective is defined by two linear equations with inequality and a single equation with equality, is depicted in Figure 6.5.

We write a system of linear equations with equalities and inequalities in a matrix form. A system of linear equations

$$\begin{aligned} m_{11}x_1 + m_{21}x_2 + \ldots + m_{d1}x_d &\sim_1 b_1, \\ &\vdots \\ m_{1n}x_1 + m_{2n}x_2 + \ldots + m_{dn}x_d &\sim_n b_n, \end{aligned}$$

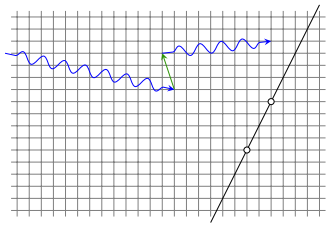


Figure 6.5: An illustration of a two-dimensional k-control $\mathbb{Z}$-VAS game, where the objective is defined by a system of linear equations with equalities and inequalities.

where $\sim_i \in \{=, \neq, <, >\}$ for all $i = 1, \ldots, n$, can be written as $M \in \mathbb{Z}^{d \times n}$, $\mathbf{b} \in \mathbb{Z}^n$, and $\sim = (\sim_1, \ldots, \sim_n)$.

Theorem 6.10. *Consider a system of linear equations with equalities and inequalities defined by $M \in \mathbb{Z}^{d \times n}$, $\mathbf{b} \in \mathbb{Z}^n$ and $\sim \in \{=, \neq, <, >\}^n$. Deciding which player has a winning strategy in a k-control $\mathbb{Z}^d$-VASS game with safety objectives, where the targets satisfy the system of linear equations, is* NP*-complete.*

Proof. Let $M \in \mathbb{Z}^{d \times n}$, $\mathbf{b} \in \mathbb{Z}^n$ and $\sim \in \{=, \neq, <, >\}^n$, be a system of linear equations with equalities and inequalities. Finding $x_1, \ldots, x_d \in \mathbb{Z}$ satisfying these equations is a well-known NP-complete problem. Incorporating solving this system of linear equations into Algorithm 6.2 does not increase the complexity and thus the problem is in NP. Hardness follows by the same arguments as in the previous section. That is, by considering a strategy of Adam where he only skips. □

On the other hand, if the target set is defined by a semi-linear set, the complexity grows significantly. The problem of checking whether $\mathbf{x}$ is in a given semi-linear set is in double exponential time [69], which increases the complexity of the whole algorithm.

6.3 Reachability of k-control $\mathbb{Z}^d$-VASS games

In this section, we consider the dual setting, where Eve can play her moves k times and Adam has unlimited resources. First, we present a simple lemma showing that Adam has a winning strategy in a $\mathbb{Z}^d$-VASS game where he has more moves. This is illustrated in Figure 6.6.

Lemma 6.11. *Let $(A, E, [s_0, t_0, \mathbf{x}_0])$ be a $\mathbb{Z}^d$-VASS game. If for each $s \in Q_E$ and $t \in Q_A$, $\deg(s) < \deg(t)$ and $\deg(t) > 1$, then Adam has a winning strategy.*

Proof. Let $[s, \dot{t}, \mathbf{x}]$ be a configuration of the game. Since $\deg(t) > 1$ and $\deg(t) > \deg(s)$, there exists at least one move $\langle\!\langle t, \mathbf{a}, t' \rangle\!\rangle$ such that from $[\dot{s}, t', \mathbf{x} + \mathbf{a}]$, no move of Eve results in $[s', t', \mathbf{0}]$. This is obvious as otherwise, there would exist a move of Eve $\langle\!\langle s, \mathbf{e}, s' \rangle\!\rangle \in Q_E$ such that for two moves of Adam $\langle\!\langle t, \mathbf{a}_1, t_1 \rangle\!\rangle, \langle\!\langle t, \mathbf{a}_2, t_2 \rangle\!\rangle \in Q_A$, where $\mathbf{a}_1 \neq \mathbf{a}_2$, such that $\mathbf{x} + \mathbf{a}_1 + \mathbf{e} = 0 = \mathbf{x} + \mathbf{a}_2 + \mathbf{e}$. □

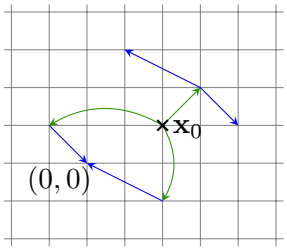


Figure 6.6: A $\mathbb{Z}^2$-VAS game, where Adam has moves $(-3, 0)$, $(0, -2)$ and $(1, 1)$ and Eve has moves $(1, -1)$ and $(-2, 1)$.

Before considering $\mathbb{Z}^d$-VASS games, we consider stateless games and prove that the winner can be decided in polynomial time. We prove the claim by showing that Eve can reach only finite number of points in $\mathbb{Z}^d$, that is, the arena is finite.

Theorem 6.12. *Deciding which player has a winning strategy in k-control $\mathbb{Z}^d$-VAS games with a reachability objective is in* PTIME.

Proof. We observe that only a finite area needs to be considered. Indeed, by a similar argument as in Lemma 6.11, Adam can avoid the origin if Eve played Skip. That is, after Eve has played all k moves, Adam has a trivial winning strategy.

If we only consider effect of Eve's moves during a play, then there are at most $\binom{m+1+k-1}{k}$ vectors she can reach. We denote this number by q. It is clear that Adam wins as soon as he can reach more than q vectors. Due to commutativity of vector addition, the arena is finite and polynomial in size of the input. We can compute whether the initial vector is winning for Eve in polynomial time using the standard attractor construction. □

It is easy to see that, by guessing the state transitions of the players', deciding which player has a winning strategy in a reachability k-control $\mathbb{Z}^d$-VASS game is in NP. This contrasts the safety k-control games where there was no complexity difference between the variant with states and without states.

Theorem 6.13. *Deciding which player has a winning strategy in reachability k-control $\mathbb{Z}^d$-VAS games is in* NP.

6.4 Concluding remarks and open problems

In this chapter, we considered k-control $\mathbb{Z}^d$-VASS games with limited resources. If we limit the available resources of Adam by allowing him to only play at most k times, then the problem of deciding the winner is NP-complete. On the other hand, if we limit number of times Eve is allowed to play, then the problem is in PTIME for stateless games and in NP for games with states.

The k-control VAS games remain open for future research. The reachability problem for VAS is a hard problem that has been known to be decidable [118] for over 35 years, and only recently Leroux and Schmitz provided the first known upper bound [109]. Also the corresponding k-control game should be challenging.

In our considerations, the number of Adam's moves, k, is not part of the input. If k is a part of the input, then the algorithms for k-control $\mathbb{Z}^d$-VAS games and k-control $\mathbb{Z}^d$-VASS games with states are exponential in contrast to general $\mathbb{Z}^2$-VAS games, where checking for existence of a winning strategy is undecidable.

Chapter 7

Single-player reachability games

In this chapter, we shift our focus from two-player dynamics to one-player systems. While most of the thesis is about different two-player games, it is vital to understand the behaviour of the system with a single player. Consider for example braid games on B_3 of subsection 3.2.5. We showed that checking for existence of a winning strategy is an undecidable problem. On the other hand, if Adam can only play a trivial braid, then in this single-player game, it is decidable in NP whether the trivial braid can be reached from the trivial braid and NP-complete from a given braid [96].

We consider two one-player systems. First, we consider the reachability problem for register machines with polynomial updates and after that the identity problem for matrices. Polynomial register machines are a generalisation of VASS and related models such as VASS with resets [63] and $\mathbb{Z}^d$-VASS [77]. Consider the operations on the counter in a VASS or a $\mathbb{Z}$-VASS, they are all of the form $x = x + a$ for some $a \in \mathbb{Z}$, while the resets are of the form $x = 0 \cdot x$. In polynomial register machines, the update function can be any polynomial with integer coefficients.

More precisely, we consider the reachability problem for polynomial iteration. That is, we are given a set of integer valued polynomials $\{p_1(x), p_2(x), \ldots, p_n(x)\} \subseteq \mathbb{Z}[x]$, two integers x_0 and x_f, and are asked whether there exists a finite sequence of polynomials $p_{i_1}(x), p_{i_2}(x), \ldots, p_{i_j}(x)$ that maps x_0 to x_f, i.e., whether

$$p_{i_j}(p_{i_{j-1}}(\cdots p_{i_2}(p_{i_1}(x_0))\cdots) = x_f.$$

The problem can be seen as a special case of *polynomial register machines* of [68], where

the machine has a single state. In [68], it was proved that the reachability problem is PSPACE-complete. We show that the reachability problem for polynomial iteration has the same complexity. The upper bound follows naturally from [68] and we highlight the authors' observations regarding this surprising complexity. The first observation is that the reachability set is not semi-linear, which can be seen by considering the polynomial $p(x) = x^2$ and its reachability set. The second observation is that the representation of evaluations of x grows exponentially with the number of times a polynomial is applied.

Example 7.1. Let $\mathcal{P} = \{x^2+x+3, x^4+2x^3+3x^2+2x+1, -x+5\}$ be the set of polynomials, $x_0 = 6$ be the initial integer and $x_f = 0$ be the target integer. Denote $p_1(x) = x^2 + x + 3$, $p_2(x) = x^4 + 2x^3 + 3x^2 + 2x + 1$ and $p_3(x) = -x + 5$. We see that, for these polynomials, x_f is reachable from x_0 as $p_3(p_1(p_2(p_3(6)))) = 0$. This iteration is depicted in Figure 7.1

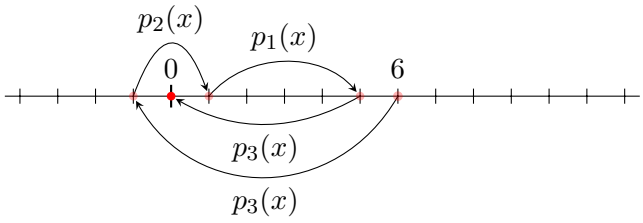


Figure 7.1: Polynomial iteration.

To prove the lower bound, we need to modify the proof of PSPACE-hardness for register machines with polynomial updates to remove the need for the state structure of the machine. To this end, we follow the proof of [68] and reduce the reachability problem for linear-bounded automata to polynomial iteration. Linear-bounded automata were studied in [106, 107], where the authors showed that the linear-bounded automata accept exactly context-sensitive languages. A linear-bounded automaton is a Turing machine with a finite tape whose length is bounded by a linear function of the length of the input. The reachability problem for the linear-bounded automaton is to decide whether starting from the initial state and the read/write head in the leftmost cell of the empty tape, the machine can reach a final state with the empty tape. The length of the tape is n and is part of the input. The problem is a well-known PSPACE-complete problem. We use an encoding similar to the encoding used in [68], where the tape content of a linear-bounded automaton was encoded as a solution to a system of linear congruences and the state structure was encoded as a state structure of a polynomial register machine. Unlike Finkel's, Göller's and

Haase's encoding, we will also encode the state structure of the linear-bounded automaton as a solution to additional linear congruences making our automaton stateless.

We also consider iterating multidimensional polynomials and show that the reachability problem is undecidable for three-dimensional polynomials. The result is not surprising as the reachability problem for register machines with two-dimensional affine updates is undecidable [137]. Unfortunately, we are not able to obtain as strong a result for polynomial iteration as, in our construction, simulating the state structure requires an additional dimension and polynomials of higher degree than one.

In the second section, we consider the identity problem for matrix semigroups. This model is similar to the single-player variant of matrix games on vectors of subsection 3.2.4, but rather than considering whether a given vector can be transformed into a target vector using provided matrices, we are asked whether the identity matrix can be constructed via matrix multiplication.

First, in analogy to a result from 1999 on non-existence of embedding into 2×2 matrix semigroups [36], we expand a horizon of the decidability area for matrix semigroups and show that there is no embedding from a set of pairs of words over a semigroup alphabet to any matrix semigroup in $\mathrm{SL}(3,\mathbb{Z})$ and in $\mathrm{H}(3,\mathbb{C})$. From the first result it follows almost immediately that there is no embedding from a set of pairs of group words into $\mathbb{Z}^{3\times 3}$.[12] The matrix semigroup in $\mathrm{SL}(3,\mathbb{Z})$ has attracted a lot of attention recently as it can be represented by a set of generators and relations [55, 56] similar to $\mathrm{SL}(2,\mathbb{Z})$, where it was possible to convert numerical problems into symbolic problems and solve them with novel computational techniques, see [11, 45, 134, 135]. Comparing to the relatively simple representation of $\mathrm{SL}(2,\mathbb{Z})$, the case of $\mathrm{SL}(3,\mathbb{Z})$ looks more challenging, as it contains many types of non-commutative and partially commutative elements.

As the decidability status of the identity problem in dimension three is a long standing open problem, we look for a subclass of $\mathrm{SL}(3,\mathbb{Z})$ for which the identity problem could be decidable following our result on existence of embeddings. The Heisenberg group is an important subgroup of $\mathrm{SL}(3,\mathbb{Z})$ which is useful in the description of one-dimensional quantum mechanical systems [32, 74, 100]. We show that the identity problem for a matrix semigroup generated by matrices from $\mathrm{H}(3,\mathbb{Q})$ is decidable in polynomial time. Furthermore,

[12] The idea that such a result may hold was motivated by analogy from combinatorial topology, where the identity problem is decidable for the braid group B_3 which is the universal central extension of the modular group $\mathrm{PSL}(2,\mathbb{Z})$ [133], the embedding for a set of pairs of words into the braid group B_5 exists, see [18], and non-existence of embeddings were proved for B_4 in [4]. So $\mathrm{SL}(3,\mathbb{Z})$ was somewhere in the goldilocks zone between B_3 and B_5.

we extend the decidability result to $\mathrm{H}(d, \mathbb{Q})$ and show that the problem is still solvable in any dimension. As the identity problem is computationally equivalent to the subgroup problem all above results hold for the subgroup problem as well.

Moreover, we fill the gap between decidability and undecidability results by improving the first undecidability result for the identity problem by substantially reducing the bound on the size of the generator set from 48 to eight for 4×4 matrix semigroups over integers. The better bound is achieved by developing a novel reduction technique which exploits the properties of anti-diagonal coordinates, in contrast to the previous technique, where a computation is repeated over several disjoint group alphabets [14].

7.1 Iterating polynomials

In this section, we prove that the reachability problem for iterating polynomials is PSPACE-complete. The proof of the lower bound is similar to the proof of PSPACE-hardness of the reachability problem for polynomial register machines. Both proofs reduce from the reachability problem for LBA. Let us fix an LBA $\mathcal{M}$ with a tape of n letters for the remainder of the section. The main difference of the proofs is that in [68], the states of PRM contain partial information from a configuration of $\mathcal{M}$: the state of $\mathcal{M}$, the position of the read/write head and the letter that the head is currently reading, while the tape content is encoded as an integer and modified by the transitions according to the instructions of $\mathcal{M}$. In our proof, the whole configuration of $\mathcal{M}$ is encoded as an integer and updated according to the instructions of $\mathcal{M}$.

First, let us recall some definitions from [68]. Let p_i denote the $(i+3)$-th prime number, that is, $p_1 = 7, p_2 = 11, \ldots$ and let P be the product of m such primes, $P = \prod_{i=1}^{m} p_i$, where $m = n + n \cdot |Q|$.

The main idea of the encoding is to consider the integer line $\mathbb{Z}$ modulo P and integers as the corresponding residue classes. We are interested in the residue classes that satisfy linear congruences modulo p_i for different p_i. The first n primes will correspond to each cell of the tape and the next $n \cdot |Q|$ primes will correspond to the head being in a particular state in a particular cell. Note, that for the sake of simplicity, we omit the behaviour of the head on the border letters. In fact, it is quite easy to deal with them as, among other information, we also encode the position of the head into our integer. Then it is easy to hard-code the behaviour of $\mathcal{M}$ on the border letters into corresponding polynomials. Indeed, consider a configuration $[q, 1, \triangleright w \triangleleft]$ such that the next configuration is $[q', 0, \triangleright w' \triangleleft]$, where

$w[i] = w'[i]$ for $i = 2, \ldots, n$. As the head respects the border letters, the next configuration is $[q'', 1, \triangleright w' \triangleleft]$. That is, we need to simulate a transition from state q to q'' and possible rewriting of the first letter of w.

We are not interested in all residue classes modulo P and only a tiny fraction of the residue classes is used to store information. A residue class r is of interest to us, if for every $1 \leq i \leq m$, there is some $b_i \in \{0, 1, 2\}$ such that $r \equiv b_i \mod p_i$. We call such residue class *sane* and denote the set of all sane residue classes by S. A configuration $[q_j, i, \triangleright w \triangleleft]$, where $i = \{1, \ldots, n\}$ and $w \in \{0, 1\}^n$, corresponds to a residue class r satisfying the system of congruence equations

$$\begin{aligned} r &\equiv w[1] \mod p_1, \\ r &\equiv w[2] \mod p_2, \\ &\vdots \\ r &\equiv w[n] \mod p_n, \end{aligned} \qquad \begin{aligned} r &\equiv 1 \mod p_\ell \text{ if } \ell = n + j + (i-1)|Q|, \\ r &\equiv 0 \mod p_\ell \text{ if } \ell > n \text{ and } \ell \neq n + j + (i-1)|Q|. \end{aligned} \tag{7.1}$$

Intuitively, the first n congruence equations encode the tape content and the next $n|Q|$ equations encode the state and position of the head. We illustrate how a configuration $[q_3, 2, \triangleright 1001 \cdots 1 \triangleleft]$ of an LBA corresponds to the residue class r satisfying the system of linear congruences (7.1) in Figure 7.2.

Since the head of an LBA modifies the tape locally, to simulate a transition, for example, $\delta(q_j, a) = (q_k, a', L)$, it is enough to check that the residue class r satisfies congruence equations

$$r \equiv 1 \mod p_{n+j+(i-1)|Q|} \qquad \text{and} \qquad r \equiv a \mod p_i$$

for some $i \in \{1, \ldots, n\}$. Then, the transition is simulated by moving from r to a residue class r' satisfying congruence equations

$$\begin{aligned} r' &\equiv 0 \mod p_{n+j+(i-1)|Q|}, \\ r' &\equiv 1 \mod p_{n+k+(i-2)|Q|}, \\ r' &\equiv a' \mod p_i, \\ r' &\equiv r \mod p_\ell \text{ for all} \\ &\qquad \ell \in \{1, \ldots, n + n \cdot |Q|\} \setminus \{i, n + j + (i-1)|Q|, n + k + (i-2)|Q|\}. \end{aligned}$$

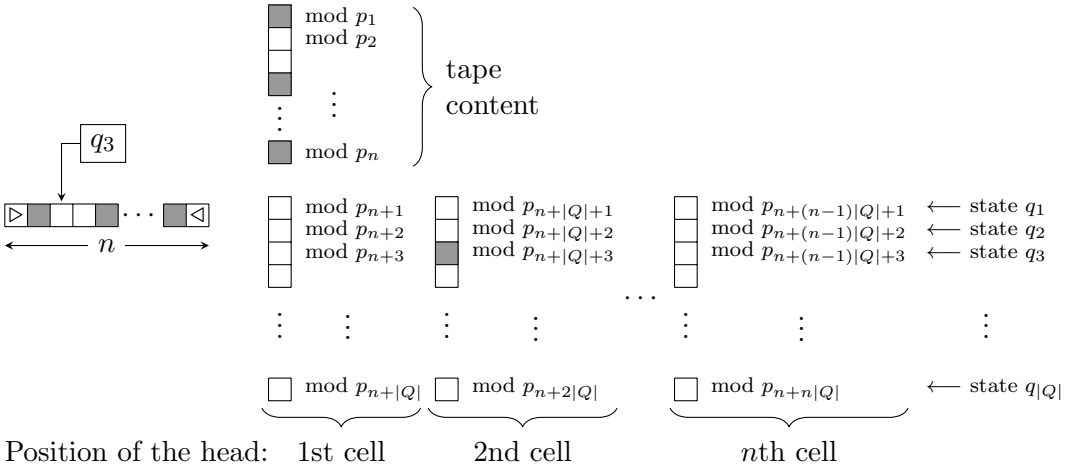


Figure 7.2: An illustration how configuration $[q_3, 2, \triangleright 1001 \cdots 1 \triangleleft]$ of an LBA (left) is encoded as residue class r satisfying a system of linear congruences. Here, letters 0 and 1 are represented by white and grey squares, respectively. A grey square in the ith cell column and the jth state row represents the head being in the ith cell in state q_j.

That is, to simulate a transition of the LBA, first we need to check that the current residue class r corresponds to a configuration $[q_j, i, \triangleright w \triangleleft]$, where $w[i] = a$, for some i and the other letters of w are irrelevant. Then we move to the residue class r' corresponding to the configuration $[q_k, i-1, \triangleright w' \triangleleft]$, where $w'[i] = a'$ and $w'[\ell] = w[\ell]$, for all $\ell = \{1, \ldots, n\} \setminus \{i\}$. Analogously, to simulate a transition where the head moves to the right, similar checks need to be performed.

To this end, we need to *locally* modify the residue classes. That is, we need to have a polynomial $p(x)$ such that $p(r) = r'$. There are three mappings that are defined for each index $i \in \{1, \ldots, m\}$, $\textsc{flip}_i, \textsc{eqzero}_i, \textsc{eqone}_i : S \to S$. The mapping $\textsc{flip}_i$ is used to change the value on the right-hand side in the congruence of p_i, the mappings $\textsc{eqzero}_i$ and $\textsc{eqone}_i$ check that on the right-hand side of the congruence of p_i is zero or one, respectively.

Let us describe the mappings in more details. For the mapping $\textsc{flip}_i(r)$ there are three cases depending on whether $r \equiv 0, 1, 2 \mod p_i$:

$$\text{if } r \equiv 0 \mod p_i: \quad \textsc{flip}_i(r) \equiv \begin{cases} 1 & \mod p_i \\ r & \mod p_j \end{cases} \qquad \text{if } r \equiv 1 \mod p_i: \quad \textsc{flip}_i(r) \equiv \begin{cases} 0 & \mod p_i \\ r & \mod p_j \end{cases}$$

$$\text{if } r \equiv 2 \mod p_i: \quad \textsc{flip}_i(r) \equiv \begin{cases} 2 & \mod p_i \\ r & \mod p_j, \end{cases}$$

where $j \neq i$.

Similarly, for the remaining two mappings, there are three cases depending on whether $r \equiv 0, 1, 2 \mod p_i$:

$$\text{if } r \equiv 0 \mod p_i: \qquad \text{EQZERO}_i(r) \equiv \begin{cases} 0 & \mod p_i \\ r & \mod p_j \end{cases}$$

$$\text{if } r \equiv 1, 2 \mod p_i: \qquad \text{EQZERO}_i(r) \equiv \begin{cases} 2 & \mod p_i \\ r & \mod p_j \end{cases}$$

$$\text{if } r \equiv 1 \mod p_i: \qquad \text{EQONE}_i(r) \equiv \begin{cases} 1 & \mod p_i \\ r & \mod p_j \end{cases}$$

$$\text{if } r \equiv 0, 2 \mod p_i: \qquad \text{EQZERO}_i(r) \equiv \begin{cases} 2 & \mod p_i \\ r & \mod p_j, \end{cases}$$

where $j \neq i$.

The move $\delta(q_j, 0) = (q_k, 0, L)$ of LBA $\mathcal{M}$ when the head is in ith position is now realised by a composition of the functions

$$\text{FLIP}_{n+k+(i-2)|Q|} \circ \text{FLIP}_{n+j+(i-1)|Q|} \circ \text{EQZERO}_i \circ \text{EQONE}_{n+j+(i-1)|Q|}.$$

In Figure 7.3 we illustrate how moves $\delta(q_j, 0) = (q_k, 0, L)$ and $\delta(q_j, 1) = (q_k, 0, R)$ of LBA $\mathcal{M}$ are realised for the configuration $[q, i, \triangleright w \triangleleft]$. Note, that we do not assume that $q = q_j$ or that $w[i] = 0$ for the first move or $w[i] = 1$ for the second move. Moves EQZERO_ℓ and EQONE_ℓ verify that the bit encoded in the residue class modulo p_ℓ is 0 or 1, respectively.

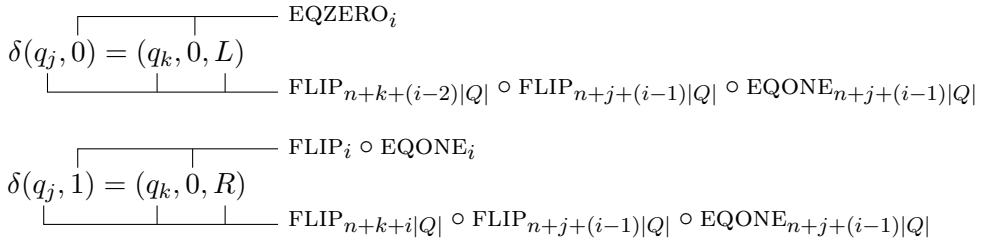


Figure 7.3: An illustration of mappings corresponding to moves of LBA.

The crucial ingredient for the simulation is that the functions FLIP_i, EQZERO_i and EQONE_i can be realised by polynomials with coefficients in $\{0, \ldots, P-1\}$. We present the lemma of [68].

Lemma 7.2. *For any $1 \leq i \leq m$ and any of $\text{FLIP}_i, \text{EQZERO}_i, \text{EQONE}_i : S \to S$, there is a quadratic polynomial with coefficients from $\{0, \ldots, P-1\}$ that realises the respective function.*

Proof. First, we show the polynomials corresponding to the mappings FLIP_i, EQZERO_i and EQONE_i that map the values correctly when considering only $\mathbb{Z}/p_i\mathbb{Z}$. Then we mention how to modify them to also map the values correctly for all $\mathbb{Z}/p_j\mathbb{Z}$, where $j \neq i$.

It is easy to verify that the polynomials

$$\begin{aligned} p_{eqzero}(x) &= -x^2 + 3x, \\ p_{eqone}(x) &= x^2 - 2x + 2 \text{ and} \\ p_{flip}(x) &= 3 \cdot 2^{-1}x^2 - 5 \cdot 2^{-1}x + 1 \end{aligned}$$

realise the respective mappings. Note that since $p_i \geq 7$, 2 has a multiplicative inverse. For example, let $p_i = 11$, then $2^{-1} = \frac{p_i - 1}{2} = 6$ and the evaluations of polynomials $p_{eqzero}(x)$, $p_{eqone}(x)$ and $p_{flip}(x)$ are presented in Table 7.1.

x	$p_{eqzero}(x)$	$p_{eqone}(x)$	$p_{flip}(x)$
0	$-0^2 + 3 \cdot 0 \equiv \mathbf{0}$	$0^2 - 2 \cdot 0 + 2 \equiv \mathbf{2}$	$3 \cdot 6 \cdot 0^2 - 5 \cdot 6 \cdot 0 + 1 \equiv \mathbf{1}$
1	$-1^2 + 3 \cdot 1 \equiv \mathbf{2}$	$1^2 - 2 \cdot 1 + 2 \equiv \mathbf{1}$	$3 \cdot 6 \cdot 1^2 - 5 \cdot 6 \cdot 1 + 1 = -11 \equiv \mathbf{0}$
2	$-2^2 + 3 \cdot 2 \equiv \mathbf{2}$	$2^2 - 2 \cdot 2 + 2 \equiv \mathbf{2}$	$3 \cdot 6 \cdot 2^2 - 5 \cdot 6 \cdot 2 + 1 = 13 \equiv \mathbf{2}$

Table 7.1: Evaluations of polynomials $p_{eqzero}(x)$, $p_{eqone}(x)$ and $p_{flip}(x)$ in $\mathbb{Z}/11\mathbb{Z}$

Although these polynomials realise the conditions of FLIP_i, EQZERO_i and EQONE_i for i, they (generally) do not realise the conditions when $j \neq i$. That is, $p_{eqzero}(x) \neq x$ when considering the polynomials in $\mathbb{Z}/p_j\mathbb{Z}$. To illustrate this, consider $p_{eqzero}(1)$ as above, but now with respect to $p_j = 7$. By the definition of EQZERO_i, it should remain unchanged, that is $p_{eqzero}(1) = 1$ with respect to p_j. This is not the case, as 1 and 2 are different residue classes. To obtain polynomials corresponding to FLIP_i, EQZERO_i and EQONE_i, we consider polynomials $p_{eqzero}(x)$, $p_{eqone}(x)$ and $p_{flip}(x)$ as $a_2x_2 + a_1x + a_0$ and construct a system of congruences for each $\ell = \{0, 1, 2\}$:

$$\begin{aligned} x &\equiv a_\ell \mod p_i \\ x &\equiv b_\ell \mod p_j \text{ for each } j \in \{1, \ldots, m\} \setminus \{i\}, \end{aligned}$$

where $b_1 = 1$ and $b_0 = b_2 = 0$. By applying the Chinese remainder theorem, we obtain the unique solution for each coefficient and obtain the polynomials $p_{eqzero,i}(x)$, $p_{eqone,i}(x)$ and $p_{flip,i}(x)$ by replacing the original coefficients with these unique solutions. □

Now, for each $i \in \{1, \ldots, m\}$ and each transition $\delta(q_j, a) = (q_k, a', D)$, where $a, a' \in \{0, 1\}$ and $D = \{L, R\}$, there exists a polynomial of at most degree 32 realising this transition by Lemma 7.2. These polynomials are exactly our set of polynomials $\mathcal{P}$. Note, that our simulation is slightly different from [68] as there, in each step, the PRM guessed (and verified) the content of the cell where the head moves in the successive configuration and only correct moves are available due to the state structure. In our model, as there is no state structure, each time a move is simulated, we have to verify that indeed both the state and current cell are correct. The initial value x_0 satisfies

$$x_0 \equiv 1 \mod p_{n+1} \qquad \text{and} \qquad x_0 \equiv 0 \mod p_\ell \text{ if } \ell \neq n+1.$$

The main idea is still the same, if $\mathcal{M}$ is simulated incorrectly, the value x becomes 2 modulo some prime p_ℓ and will remain 2 forever.

It should be also pointed out that simulating behaviour on the border of the tape is easy and results in polynomials of at most degree 32. Moreover, the sequence of configurations $[q_j, 1, \triangleright w \triangleleft] \to [q_k, 0, \triangleright w' \triangleleft] \to [q_j, 1, \triangleright w' \triangleleft]$ does not require any special considerations as each polynomial contains EQONE_{n+j} ensuring that this sort of loop cannot be used to unfaithfully modify the first letter on the tape.

By induction on the length of the run of LBA $\mathcal{M}$, it is easy to see that $[q_1, 0, \triangleright 0^n \triangleleft] \to^* [q_f, 0, \triangleright 0^n \triangleleft]$ in $\mathcal{M}$ if and only if a residue class r, such that

$$r \equiv 1 \mod p_{n+|Q|} \qquad \text{and} \qquad r \equiv 0 \mod p_\ell \text{ if } \ell \neq n + |Q|,$$

is reachable from x_0 by applying polynomials from $\mathcal{P}$. To reach 0, we need three additional polynomials: one polynomial to move to a residue class r' such that $r' \equiv 0 \mod p_\ell$ for all $1 \leq \ell \leq m$, and two polynomials to move from the *integer* r' to 0. The first polynomial is $p_{flip,n+|Q|}(p_{eqone,n+|Q|}(x))$ as we assumed that the final state appears only in the configuration $[q_{|Q|}, 0, \triangleright 0^n \triangleleft]$. The latter polynomials are $p_+(x) = x + P$ and $p_-(x) = x - P$.

We have proved the following lemma:

Lemma 7.3. *The reachability problem for polynomial iteration is* PSPACE*-hard for polynomials with integer coefficients.*

We illustrate the simulation of an LBA with polynomials.

Example 7.4. Let $\mathcal{M}$ be an LBA with a single state, a tape with two cells and a move $\delta(q_1, 0) = (q_1, 1, R)$. We need two primes for the tape and another two for the head. That is, we use primes $7, 11, 13$ and 17. For the sake of readability, we present all the integers modulo $P = 7 \cdot 11 \cdot 13 \cdot 17 = 17017$. The integers r and s representing configurations $[q_1, 1, \triangleright 00 \triangleleft]$ and $[q_1, 2, \triangleright 10 \triangleleft]$ can be solved from the system of congruences

$$\begin{aligned} r &\equiv 0 \mod 7, & s &\equiv 1 \mod 7 \\ r &\equiv 0 \mod 11, & s &\equiv 0 \mod 11, \\ r &\equiv 1 \mod 13, & s &\equiv 0 \mod 13, \\ r &\equiv 0 \mod 17, & s &\equiv 1 \mod 17 \end{aligned}$$

using the Chinese remainder theorem. That is, $r = 3927$ and $s = 715$. The move $\delta(q_1, 0) = (q_1, 1, R)$ is realised by $\text{FLIP}_1 \circ \text{EQZERO}_1 \circ \text{FLIP}_4 \circ \text{FLIP}_3 \circ \text{EQONE}_3$. By Lemma 7.2, EQONE_3 is realised by a quadratic polynomial $a_2'x^2 + a_1'x + a_0'$ with coefficients satisfying the congruences

$$\begin{aligned} a_2' &\equiv 0 \mod 7, & a_1' &\equiv 1 \mod 7, & a_0' &\equiv 0 \mod 7, \\ a_2' &\equiv 0 \mod 11, & a_1' &\equiv 1 \mod 11, & a_0' &\equiv 0 \mod 11, \\ a_2' &\equiv 1 \mod 13, & a_1' &\equiv -2 \mod 13, & a_0' &\equiv 2 \mod 13, \\ a_2' &\equiv 0 \mod 17, & a_1' &\equiv 1 \mod 17, & a_0' &\equiv 0 \mod 17. \end{aligned}$$

Solving these systems using the Chinese remainder theorem, we see that $p_{eqone,3}(x) = 3927x^2 + 5237x + 7854$. The other polynomials are solved from similar systems of congruences and are

$$\begin{aligned} p_{flip,3} &= 14399x^2 + 11782x + 3927, & p_{flip,4} &= 12012x^2 + 6007x + 8008, \\ p_{eqzero,1} &= 7293x^2 + 2432x, & p_{flip,1} &= 14586x^2 + x + 9724. \end{aligned}$$

Finally, the composition of the polynomials is $p(x) = 11968x^4 + 8041x^3 + 9207x^2 + 11056x + 8569$. It can be easily verified that $p(x)$ simulates the move $\delta(q_1, 0) = (q_1, 1, R)$ from the configuration $[q_1, 1, \triangleright 00 \triangleleft]$ correctly, i.e., $p(r) = s$.

Note, that in the previous example, the polynomial composed of five quadratic poly-

nomials is not of degree 32 but only of degree four. This is the case in general due to the coefficients of the polynomials satisfying the particular congruences. For example, consider the composition of two quadratic polynomials $\textsc{Poly}_i$ and $\textsc{Poly}'_\ell$, where $\textsc{Poly}, \textsc{Poly}' \in \{\textsc{flip}, \textsc{eqone}, \textsc{eqzero}\}$ and $i \neq \ell$. Let α, β be the coefficients of the highest power. They can be expressed as integers

$$\alpha = x \cdot \prod_{\substack{j=1,\\ j\neq i}}^{m} p_j \qquad \text{and} \qquad \beta = y \cdot \prod_{\substack{j=1,\\ j\neq \ell}}^{m} p_j,$$

where $x, y \in \mathbb{Z}$ and $i, \ell \in \{1, \ldots, m\}$ and $i \neq \ell$. Now, in the composition of $\textsc{Poly}_i$ and $\textsc{Poly}'_\ell$, the coefficient of x^4 is $\alpha \cdot \beta$, which is

$$\alpha \cdot \beta = x \cdot \prod_{\substack{j=1,\\ j\neq i}}^{m} p_j \cdot y \cdot \prod_{\substack{j=1,\\ j\neq \ell}}^{m} p_j = xy \cdot \prod_{\substack{j=1,\\ j\neq i,\ell}}^{m} p_j \cdot \prod_{j=1}^{m} p_j.$$

The coefficient is divisible by P and, thus, can be replaced by 0 in our considerations. In a similar fashion, we can show that for the polynomial $p(x)$ simulating a transition of $\mathcal{M}$, all the coefficients of degrees higher than four are divisible by P.

To prove that the reachability problem is PSPACE-complete, it remains to prove that the problem can be solved in PSPACE.

Lemma 7.5. *The reachability problem for polynomial iteration is* PSPACE *for polynomials with integer coefficients.*

Proof. Consider the set $\mathcal{P}$ as a PRM with a single state and where the transitions are labelled by the polynomials of $\mathcal{P}$. The reachability problem for PRM can be solved in PSPACE and thus also the reachability problem for polynomial iteration is in PSPACE. □

We highlight some crucial observations from the proof that the reachability problem for PRM is in PSPACE of [68]. Firstly, most of the polynomials have monotonic behaviour when integers with large absolute values are evaluated, and this absolute value is of polynomial size of the input. Secondly, the only polynomials that do not have monotonic behaviour are of the form $\pm x + b$ for some $b \in \mathbb{Z}$. They can be simulated by a one-dimensional vector addition system with states extracted from the given PRM. Moreover, simulating to which integers, with small absolute values, the polynomials of the form $\pm x + b$ can return to can be done in polynomial space.

Combining lemmas 7.3 and 7.5, we have our main result.

Theorem 7.6. *The reachability problem for polynomial iteration is* PSPACE*-complete for polynomials with integer coefficients.*

Next, we extend the previous results by considering polynomials over the field $\mathbb{Q}$. Additionally, we modify the polynomials to ensure that the image is always in $[0, 1]$.

Let $p(x)$ be a polynomial from our set $\mathcal{P}$. Then the corresponding set of rational polynomials $\mathcal{Q}$ has a polynomial $p'(x) = \frac{1}{p(\frac{1}{x})}$. It is easy to see that in fact p' is of the form $\frac{r(x)}{q(x)}$ for some $r(x), q(x) \in \mathbb{Z}[x]$. We can inherit the lower bound for the reachability for rational polynomials from Lemma 7.3.

Corollary 7.7. *The reachability problem for polynomial iteration is* PSPACE*-hard, when the polynomials are of form* $\frac{r(x)}{q(x)} : [0, 1] \to [0, 1]$ *over polynomial ring* $\mathbb{Q}[x]$*.*

Finally, we prove that the reachability problem for multidimensional polynomial iteration is undecidable. We construct three-dimensional polynomials that simulate a two-dimensional PRM with affine updates with undecidable reachability problem [137].

Theorem 7.8. *The reachability problem for multidimensional polynomial iteration is undecidable already for three-dimensional polynomials.*

Proof. Let $\mathcal{M}$ be a two-dimensional PRM with affine updates and n states. Let $Q = \{q_1, \ldots, q_n\}$ be its states and Δ the set of transitions. Without loss of generality, we can assume that q_1 is the initial state and q_n is the final state. We construct a three-dimensional set of polynomials $\mathcal{P}$ such that the first two dimensions are updated as in $\mathcal{M}$ and the third dimension is used to simulate the state transition of $\mathcal{M}$. As in the beginning of the section, let p_i be $(i+3)$-th prime number and P be the product of n primes $P = \prod_{i=1}^{n} p_i$. Intuitively, we will encode current state q_j into a residue class r satisfying $r \equiv 1 \mod p_j$ and $r \equiv 0 \mod p_\ell$ if $\ell \neq j$. Then a transition $q_j \to q_k$ is simulated by a polynomial corresponding to $\text{FLIP}_k \circ \text{FLIP}_j \circ \text{EQONE}_j$. By Lemma 7.2, such polynomials exist. More formally, for each transition $(q_j, (p_1(x), p_2(x)), q_k) \in \Delta$ the polynomial $(p_1(x), p_2(x), p_{flip,k}(p_{flip,j}(p_{eqone,j}(x))))$ is added to $\mathcal{P}$.

It is easy to see that $[q_1, (x_0, y_0)] \to^*_{\mathcal{M}} [q_n, (x_f, y_f)]$ if and only if (x_f, y_f, r_f) is reachable from (x_0, y_0, r_0), where r_0 is the residue class satisfying $r_0 \equiv 1 \mod p_1$ and $r_0 \equiv 0 \mod p_\ell$ if $\ell > 1$ and r_1 satisfies $r_f \equiv 1 \mod p_n$ and $r_f \equiv 0 \mod p_\ell$ if $\ell < n$. We add polynomials

$p_-(x, y, z) = (x, y, z-P)$, $p_+(x, y, z) = (x, y, z+P)$ and $p(x, y, z) = (x, y, p_{flip,n}(p_{eqone,n}(z)))$ to $\mathcal{P}$ to reach $(0, 0, 0)$.

The reachability problem for two-dimensional PRM with affine updates is undecidable [137] and hence so is the reachability for three-dimensional polynomial iteration. □

While we considered multidimensional polynomial iteration with univariate polynomials in each dimension, it is natural to consider multidimensional polynomials, where polynomials are really multivariate. In particular, multidimensional affine polynomials as they can be expressed as a matrix. This trade-off between degree of polynomials and dimension leads to interesting problems in matrix theory. Especially, when rather than iteration of a given vector, we consider questions regarding composition of polynomials or matrices. In case of our multidimensional univariate polynomials, composition questions are not interesting as we considered polynomials with integer coefficients where a composition of two polynomials increases the degree. But when considering multivariate polynomials, a composed polynomial degree can increase or decrease. This leads to a natural question: is there a composition of given multidimensional polynomials resulting in the identity polynomial? In terminology of matrix problems, this is the identity problem.

7.2 Non-existence of embedding from pairs of words into 3×3 matrices

Let us consider a single-player matrix multiplication game with the identity matrix as the objective. We will first look for possible domains where the reachability problem could be decidable. The identity problem was shown to be undecidable for matrices from $\mathrm{SL}(4, \mathbb{Z})$ [12]. As with most undecidability proofs for matrix problems, the main ingredient was to encode a universal computation into matrices. Roughly speaking, the proofs of undecidability rely on embedding pairs of words into matrices. In [36], the authors showed there is no embedding from a set of pairs of words into two-by-two matrices, even when the elements are complex numbers. In this section, we show that there is no embedding from a set of pairs of words over a semigroup alphabet to the special linear group $\mathrm{SL}(3, \mathbb{Z})$. This suggests that the identity problem could be decidable for 3×3 matrices.

Let $\Sigma = \{0, 1\}$. The monoid $\Sigma^* \times \Sigma^*$ has a generating set $S = \{(0, \varepsilon), (1, \varepsilon), (\varepsilon, 0), (\varepsilon, 1)\}$, where ε is the empty word. We simplify the notation by setting $a = (0, \varepsilon)$, $b = (1, \varepsilon)$,

$c = (\varepsilon, 0)$ and $d = (\varepsilon, 1)$. It is easy to see that we have the following relations:

$$ac = ca, \qquad bc = cb, \qquad ad = da, \qquad bd = db. \tag{7.2}$$

In other words, a and b commute with c and d. Furthermore, these are the only relations. That is, a and b do not commute with each other, neither do c and d, and there are no other nontrivial relations. Let $\varphi : \Sigma^* \times \Sigma^* \to \mathrm{SL}(3, \mathbb{Z})$ be an injective morphism and denote $A = \varphi(a)$, $B = \varphi(b)$, $C = \varphi(c)$ and $D = \varphi(d)$. Our goal is to show that φ does not exist. Unfortunately, the technique developed in [36], where the contradiction was derived from simple relations, resulting from the matrix multiplication, cannot be used for a case of $\mathrm{SL}(3, \mathbb{Z})$ as it creates a lot of equations which do not limit the possibility of embeddings. In contrast to [36], we found new techniques to show non-existence of φ by analysis of eigenvalues and the Jordan normal forms.

Lemma 7.9. *Let $\Sigma = \{0, 1\}$. If there is an injective morphism $\varphi : \Sigma^* \times \Sigma^* \to \mathrm{SL}(3, \mathbb{Z})$ and the matrices A, B, C and D correspond to $\varphi((0, \varepsilon))$, $\varphi((1, \varepsilon))$, $\varphi((\varepsilon, 0))$ and $\varphi((\varepsilon, 1))$ respectively, then the matrices A, B, C and D have a single eigenvalue and additionally the Jordan normal form is $\left(\begin{smallmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{smallmatrix}\right)$.*

Proof. We prove the claim by considering Jordan normal forms of matrices and showing that all but one normal form result in additional relations beside relations (7.2). Moreover, this form fixes the eigenvalues to be 1, as for other eigenvalues, the matrices are no longer in $\mathrm{SL}(3, \mathbb{Z})$.

Let φ be an injective morphism from $S = \{(0, \varepsilon), (1, \varepsilon), (\varepsilon, 0), (\varepsilon, 1)\}$ into $\mathrm{SL}(3, \mathbb{Z})$. Because of obvious symmetries, it sufficies to prove the claim for $A = \varphi((0, \varepsilon))$. Now, the only relations in $\mathrm{SL}(3, \mathbb{Z})$ are $AC = CA$, $AD = DA$, $BC = CB$ and $BD = DB$.

Since the conjugation by an invertible matrix does not influence the injectivity, we can conjugate the four matrices by some $X \in \mathbb{R}^{3\times 3}$ such that A is in the Jordan normal form. For a 3×3 matrix, there are six different types of matrices in the Jordan normal form. If A has three different eigenvalues, then

$$A = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \nu \end{pmatrix}. \tag{7.3}$$

If A has two eigenvalues, then

$$A = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \mu \end{pmatrix} \text{ or } A = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \mu & 1 \\ 0 & 0 & \mu \end{pmatrix}. \tag{7.4}$$

Finally, if A has only one eigenvalue, then

$$A = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{pmatrix} \text{ or } A = \begin{pmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{pmatrix} \text{ or } A = \begin{pmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{pmatrix}. \tag{7.5}$$

The first case (7.3) can be easily ruled out since A only commutes with diagonal matrices. Then C and D should be commuting with A by the suggested relations and as a result, C and D commute with each other.

Now let us consider the second case (7.4) where A has two eigenvalues λ and μ. Note that the determinant of A is one since $\varphi((0,\varepsilon)) \in \mathrm{SL}(3,\mathbb{Z})$. Namely, $\det(A) = \lambda\mu^2 = 1$. Also, $\mathrm{tr}(A) = \mathrm{tr}(\varphi((0,\varepsilon)) \in \mathbb{Z}$ and thus the eigenvalues are not complex numbers. Consider then the subcase, where the Jordan normal form is $\left(\begin{smallmatrix} \lambda & 0 & 0 \\ 0 & \mu & 1 \\ 0 & 0 & \mu \end{smallmatrix}\right)$ and let $C = \left(\begin{smallmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{smallmatrix}\right)$. Now

$$AC = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \mu & 1 \\ 0 & 0 & \mu \end{pmatrix} \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{pmatrix} = \begin{pmatrix} \lambda a & \lambda b & \lambda c \\ g + \mu d & h + \mu e & \ell + \mu f \\ \mu g & \mu h & \mu \ell \end{pmatrix} \text{ and}$$

$$CA = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{pmatrix} \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \mu & 1 \\ 0 & 0 & \mu \end{pmatrix} = \begin{pmatrix} \lambda a & \mu b & b + \mu c \\ \lambda d & \mu e & e + \mu f \\ \lambda g & \mu h & h + \mu \ell \end{pmatrix}.$$

Since these matrices are equal, and since $\lambda \neq \mu$, we have that $b = c = d = g = h = 0$ and $e = \ell$. Similar calculation gives us $D = \left(\begin{smallmatrix} a' & 0 & 0 \\ 0 & e' & f' \\ 0 & 0 & e' \end{smallmatrix}\right)$. Now, matrices C and D commute as follows:

$$\begin{pmatrix} a & 0 & 0 \\ 0 & e & f \\ 0 & 0 & e \end{pmatrix} \begin{pmatrix} a' & 0 & 0 \\ 0 & e' & f' \\ 0 & 0 & e' \end{pmatrix} = \begin{pmatrix} aa' & 0 & 0 \\ 0 & ee' & ef' + fe' \\ 0 & 0 & ee' \end{pmatrix} = \begin{pmatrix} a' & 0 & 0 \\ 0 & e' & f' \\ 0 & 0 & e' \end{pmatrix} \begin{pmatrix} a & 0 & 0 \\ 0 & e & f \\ 0 & 0 & e \end{pmatrix},$$

which is not one of the allowed relations.

The second subcase, where matrix A is diagonal and has two eigenvalues is ruled out in a similar fashion. First, similarly to the previous cases, we solve $C = \left(\begin{smallmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{smallmatrix}\right)$ from the equation $AC = CA$. That is,

$$AC = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \mu \end{pmatrix} \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{pmatrix} = \begin{pmatrix} \lambda a & \lambda b & \lambda c \\ \mu d & \mu e & \mu f \\ \mu g & \mu h & \mu \ell \end{pmatrix} \text{ and}$$
$$CA = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{pmatrix} \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \mu \end{pmatrix} = \begin{pmatrix} \lambda a & \mu b & \mu c \\ \lambda d & \mu e & \mu f \\ \lambda g & \mu h & \mu \ell \end{pmatrix},$$

and we have that $b = c = d = g = 0$. It is easy to see that C is diagonalisable, and thus as matrices A and C commute, they can be simultaneously diagonalised. That is, we can write

$$A = \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \mu \end{pmatrix} \text{ and } C = \begin{pmatrix} \lambda' & 0 & 0 \\ 0 & \mu' & 0 \\ 0 & 0 & \nu' \end{pmatrix},$$

where λ', μ' and ν' are the eigenvalues of C. Let $D = \left(\begin{smallmatrix} a' & b' & c' \\ d' & e' & f' \\ g' & h' & \ell' \end{smallmatrix}\right)$. It is easy to see that C and D do not commute if and only if the eigenvalues μ' and ν' are nonequal:

$$CD = \begin{pmatrix} \lambda' & 0 & 0 \\ 0 & \mu' & 0 \\ 0 & 0 & \nu' \end{pmatrix} \begin{pmatrix} a' & b' & c' \\ d' & e' & f' \\ g' & h' & \ell' \end{pmatrix} = \begin{pmatrix} \lambda' a' & 0 & 0 \\ 0 & \mu' e' & \mu' f' \\ 0 & \nu' h' & \nu' \ell' \end{pmatrix} \text{ and}$$
$$DC = \begin{pmatrix} a' & b' & c' \\ d' & e' & f' \\ g' & h' & \ell' \end{pmatrix} \begin{pmatrix} \lambda' & 0 & 0 \\ 0 & \mu' & 0 \\ 0 & 0 & \nu' \end{pmatrix} = \begin{pmatrix} \lambda' a' & 0 & 0 \\ 0 & \mu' e' & \nu' f' \\ 0 & \mu' h' & \nu' \ell' \end{pmatrix}.$$

But if C has three distinct eigenvalues λ', μ' and ν', then as when we considered case (7.3), matrix B is diagonal and hence commutes with A.

Finally, we consider the case (7.5) where A has only one eigenvalue. In the first subcase, the matrix A is diagonal and it is easy to see that then A commutes with all matrices, including B.

Let us then consider the second subcase, where the matrix A is in the following form $\left(\begin{smallmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{smallmatrix}\right)$ and let $C = \left(\begin{smallmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{smallmatrix}\right)$. Now

$$AC = \begin{pmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{pmatrix} \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{pmatrix} = \begin{pmatrix} d + a\lambda & e + b\lambda & f + c\lambda \\ g + d\lambda & h + e\lambda & \ell + f\lambda \\ g\lambda & h\lambda & \ell\lambda \end{pmatrix} \text{ and}$$

$$CA = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & \ell \end{pmatrix} \begin{pmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 1 \\ 0 & 0 & \lambda \end{pmatrix} = \begin{pmatrix} a\lambda & a + b\lambda & b + c\lambda \\ d\lambda & d + e\lambda & e + f\lambda \\ g\lambda & g + h\lambda & h + \ell\lambda \end{pmatrix}.$$

Since these matrices are equal, we have that $d = g = h = 0$, $a = e = \ell$ and $b = f$. Similar calculation gives us $D = \left(\begin{smallmatrix} a' & b' & c' \\ 0 & a' & b' \\ 0 & 0 & a' \end{smallmatrix}\right)$ and now matrices C and D commute by Lemma 2.2. Indeed, matrix C can be expressed as $C = a\left(\begin{smallmatrix} 1 & \frac{b}{a} & \frac{c}{a} \\ 0 & 1 & \frac{b}{a} \\ 0 & 0 & 1 \end{smallmatrix}\right) \in \mathrm{H}(3, \mathbb{Q})$ and matrix D has an analogous expression. Then it is clear that $\frac{b}{a}\frac{b'}{a'} = \frac{b'}{a'}\frac{b}{a}$ and thus matrices C and D commute.

Now the only possibility for A is the following form: $A = \left(\begin{smallmatrix} \lambda & 1 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{smallmatrix}\right)$, where λ is the single eigenvalue of A. Since $\det(A) = \lambda^3 = 1$ and $\mathrm{tr}(\varphi(0, \varepsilon)) = \mathrm{tr}(A) = 3\lambda$, the only solution is $\lambda = 1$. □

Based on the restriction on the Jordan normal form of matrices and number of distinct eigenvalues, we prove that there is no injective morphism from the set of pairs of words over a binary alphabet Σ into $\mathrm{SL}(3, \mathbb{Z})$.

Theorem 7.10. *There is no injective morphism $\varphi : \Sigma^* \times \Sigma^* \to \mathrm{SL}(3, \mathbb{Z})$ for any binary alphabet Σ.*

Proof. Assume to the contrary that an injective morphism φ from $\Sigma^* \times \Sigma^*$ into $\mathrm{SL}(3, \mathbb{Z})$ exists. Since the conjugation by an invertible matrix does not influence the injectivity, we suppose that the image of a is in the Jordan normal form as proven in Lemma 7.9. Let us denote the images of the generators a, b, c and d as A, B, C and D, respectively. Then we have the following matrices corresponding to the generators a, b, c and d as follows:

$$A = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} a_B & b_B & c_B \\ d_B & e_B & f_B \\ g_B & h_B & \ell_B \end{pmatrix}, \quad C = \begin{pmatrix} a_C & b_C & c_C \\ d_C & e_C & f_C \\ g_C & h_C & \ell_C \end{pmatrix}, \quad D = \begin{pmatrix} a_D & b_D & c_D \\ d_D & e_D & f_D \\ g_D & h_D & \ell_D \end{pmatrix}.$$

Note that $B, C, D \in \mathbb{R}^{3\times 3}$ as we conjugate the matrices in order for A to be in the Jordan normal form.

Since A and C commute with each other by one of the given relations in (7.2), we have

$$AC = \begin{pmatrix} a_C + d_C & b_C + e_C & c_C + f_C \\ d_C & e_C & f_C \\ g_C & h_C & \ell_C \end{pmatrix} = \begin{pmatrix} a_C & a_C + b_C & c_C \\ d_C & d_C + e_C & f_C \\ g_C & g_C + h_C & \ell_C \end{pmatrix} = CA.$$

It is easy to see that $d_C = g_C = f_C = 0$ and $a_C = e_C$. Therefore,

$$C = \begin{pmatrix} a_C & b_C & c_C \\ 0 & a_C & 0 \\ 0 & h_C & \ell_C \end{pmatrix} \text{ and } D = \begin{pmatrix} a_D & b_D & c_D \\ 0 & a_D & 0 \\ 0 & h_D & \ell_D \end{pmatrix}.$$

Since $\varphi(c)$ and $\varphi(d)$ are in $\mathrm{SL}(3,\mathbb{Z})$, the determinants of C and D are 1. Now, the determinant of C is $a_C^2 \ell_C$ and the eigenvalues are a_C and ℓ_C. As C is similar to $\varphi(c)$, the matrices have the same eigenvalues, namely the single eigenvalue is 1 by Lemma 7.9. Thus, $a_C = \ell_C = 1$. Analogously, we can also see that $a_D = \ell_D = 1$. Next, we observe that the matrices C and D commute if and only if $c_C h_D = c_D h_C$. By relations (7.2), C and D do not commute and hence there are three cases to be considered:

1. $c_C = 0$ and $h_C \neq 0$; 2. $c_C \neq 0$ and $h_C \neq 0$; 3. $c_C \neq 0$ and $h_C = 0$.

We prove that each case leads to a contradiction, i.e., that C and D commute. Let us examine the three cases in more details.

First, let us consider the case where $c_C = 0$ and $h_C \neq 0$. We know that c_D is also non-zero because otherwise C and D commute with each other since $c_C h_D = c_D h_C = 0$. We have the following calculations:

$$BC = \begin{pmatrix} a_B & b_B & c_B \\ d_B & e_B & f_B \\ g_B & h_B & \ell_B \end{pmatrix} \begin{pmatrix} 1 & b_C & 0 \\ 0 & 1 & 0 \\ 0 & h_C & 1 \end{pmatrix} = \begin{pmatrix} a_B & a_B b_C + b_B + c_B h_C & c_B \\ d_B & d_B b_C + e_B + f_B h_C & f_B \\ g_B & g_B b_C + h_B + \ell_B h_C & \ell_B \end{pmatrix} \text{ and}$$

$$CB = \begin{pmatrix} 1 & b_C & 0 \\ 0 & 1 & 0 \\ 0 & h_C & 1 \end{pmatrix} \begin{pmatrix} a_B & b_B & c_B \\ d_B & e_B & f_B \\ g_B & h_B & \ell_B \end{pmatrix} = \begin{pmatrix} a_B + d_B b_C & b_B + e_B b_C & c_B + f_B b_C \\ d_B & e_B & f_B \\ d_B h_C + g_B & e_B h_C + h_B & f_B h_C + \ell_B \end{pmatrix}.$$

Since $BC = CB$, we have $d_B b_C = 0$, $d_B h_C = 0$, $f_B b_C = 0$, and $f_B h_C = 0$. By the supposition $h_C \neq 0$, we further deduce that $d_B = f_B = 0$. Then B is $B = \begin{pmatrix} a_B & b_B & c_B \\ 0 & e_B & 0 \\ g_B & h_B & \ell_B \end{pmatrix}$. Note that we also have

$$a_B b_C + c_B h_C = e_B b_C \text{ and } g_B b_C + \ell_B h_C = e_B h_C \tag{7.6}$$

by the equality $BC = CB$.

The characteristic polynomial of B is

$$P(x) = -x^3 + \mathrm{tr}(B)x^2 - (a_B e_B + a_B \ell_B + e_B \ell_B - c_B g_B)x + \det(B)$$

which has roots $\lambda = e_B$ and $\lambda = \frac{1}{2}(a_B + \ell_B \pm \sqrt{(a_B - \ell_B)^2 + 4c_B g_B})$. We know that the only eigenvalue of B is 1 by Lemma 7.9 and therefore, we have $a_B = e_B = \ell_B = 1$ and $c_B g_B = 0$.

Moreover, it follows from equation (7.6) that $c_B = 0$ and $g_B b_C = 0$. Note that $g_B \neq 0$ because otherwise the matrix B commutes with A. Finally, we consider

$$BD = \begin{pmatrix} 1 & b_B & 0 \\ 0 & 1 & 0 \\ g_B & h_B & 1 \end{pmatrix} \begin{pmatrix} 1 & b_D & c_D \\ 0 & 1 & 0 \\ 0 & h_D & 1 \end{pmatrix} = \begin{pmatrix} 1 & b_B + b_D & c_D \\ 0 & 1 & 0 \\ g_B & g_B b_D + h_B + h_D & g_B c_D + 1 \end{pmatrix}$$
$$DB = \begin{pmatrix} 1 & b_D & c_D \\ 0 & 1 & 0 \\ 0 & h_D & 1 \end{pmatrix} \begin{pmatrix} 1 & b_B & 0 \\ 0 & 1 & 0 \\ g_B & h_B & 1 \end{pmatrix} = \begin{pmatrix} c_D g_B + 1 & b_B + b_D + c_D h_B & c_D \\ 0 & 1 & 0 \\ g_B & h_B + h_D & 1 \end{pmatrix}.$$

It is easy to see that $b_D = c_D = 0$ and then D commutes with C. Therefore, we have a contradiction.

Let us consider the second case where $c_C \neq 0$ and $h_C = 0$. It is quite similar to the previous case. Consider the products BC and CB as follows:

$$BC = \begin{pmatrix} a_B & b_B & c_B \\ d_B & e_B & f_B \\ g_B & h_B & \ell_B \end{pmatrix} \begin{pmatrix} 1 & b_C & c_C \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} a_B & a_B b_C + b_B & a_B c_C + c_B \\ d_B & d_B b_C + e_B & d_B c_C + f_B \\ g_B & g_B b_C + h_B & g_B c_C + \ell_B \end{pmatrix} \text{ and}$$
$$CB = \begin{pmatrix} a_B + d_B b_C + g_B c_C & b_B + e_B b_C + h_B c_C & c_B + f_B b_C + \ell_B c_C \\ d_B & e_B & f_B \\ g_B & h_B & \ell_B \end{pmatrix}.$$

Since the matrix B commutes with C, we have $d_B b_C = 0$, $g_B b_C = 0$, $g_B c_C = 0$, and

$d_B c_C = 0$. By the supposition $c_C \neq 0$, we further deduce that $d_B = g_B = 0$. Then B is of the following form: $B = \begin{pmatrix} a_B & b_B & c_B \\ 0 & e_B & f_B \\ 0 & h_B & \ell_B \end{pmatrix}$. Note that we also have

$$a_B b_C = e_B b_C + h_B c_C \text{ and } a_B c_C = f_B b_C + \ell_B c_C \tag{7.7}$$

by the equality $BC = CB$.

The characteristic polynomial of B is $P(x) = -x^3 + \mathrm{tr}(B)x^2 - (a_B e_B + a_B \ell_B + e_B \ell_B - f_B h_B)x + \det(B)$ which has roots $\lambda = e_B$ and $\lambda = \frac{1}{2}(e_B + \ell_B \pm \sqrt{(a_B - \ell_B)^2 + 4 f_B h_B})$. We know that the only eigenvalue of B is 1 by Lemma 7.9 and therefore, we have $a_B = e_B = \ell_B = 1$ and $f_B h_B = 0$.

We can further deduce from equation (7.7) that $h_B = 0$ and $f_B b_C = 0$. By a similar argument for the matrices B and D that should commute with each other as in the first case, we have a contradiction.

Finally, consider the third case where $c_C \neq 0$ and $h_C \neq 0$. It is obvious that c_D and h_D are also non-zero because otherwise C and D would commute. Now consider the matrix B which is commuting with C and D. We can deduce from the relation $BC = CB$ that $d_B = g_B = f_B = 0$ and $a_B = e_B = \ell_B = 1$ since they are eigenvalues of B. Hence, $B = \begin{pmatrix} 1 & b_B & c_B \\ 0 & 1 & 0 \\ 0 & h_B & 1 \end{pmatrix}$.

Now we have $c_C h_B = c_B h_C$ since B and C commute with each other. Note that h_B and c_B are both non-zero since A and B commute if $h_B = c_B = 0$. Let us denote $\frac{c_C}{h_C} = \frac{c_B}{h_B} = x$. We also have $c_D h_B = c_B h_D$ from the relation $BD = DB$ and have $\frac{c_D}{h_D} = \frac{c_B}{h_B} = x$. From $x = \frac{c_C}{h_C} = \frac{c_D}{h_D}$, we have $c_C h_D = c_D h_C$ which results in the relation $CD = DC$. Therefore, we also have a contradiction.

Since we have examined all possible cases and found contradictions for every case, we can conclude that there is no injective morphism from $\Sigma^* \times \Sigma^*$ into the special linear group $\mathrm{SL}(3, \mathbb{Z})$. □

Corollary 7.11. *There is no injective morphism $\varphi : \mathrm{FG}(\Gamma) \times \mathrm{FG}(\Gamma) \to \mathbb{Z}^{3\times 3}$ for any binary group alphabet Γ.*

Proof. We proceed by contradiction. Assume that there exists such an injective morphism φ from the set of pairs of words over a group alphabet to the set of matrices in $\mathbb{Z}^{3\times 3}$. Suppose that $A = \varphi(a, \varepsilon)$, where $a \in \Sigma$. Then the inverse matrix A^{-1} corresponding to $(\bar{a}, \varepsilon)$ must be in $\mathbb{Z}^{3\times 3}$. This implies that the determinant of A is 1 because otherwise the determinant

of A^{-1} becomes a non-integer. By Theorem 7.10, such injective morphism φ does not exist. □

Next, we show that there does not exist an embedding from pairs of words over a semigroup alphabet into matrices from $\mathrm{H}(3,\mathbb{C})$.

Theorem 7.12. *There is no injective morphism $\varphi : \Sigma^* \times \Sigma^* \to \mathrm{H}(3,\mathbb{C})$ for any binary alphabet Σ.*

Proof. Assume to the contrary that there is an injective morphism φ from $\Sigma^* \times \Sigma^*$ into $\mathrm{H}(3,\mathbb{C})$. Using the notations and relations of (7.2), we set $\varphi(a) = A$, $\varphi(b) = B$, $\varphi(c) = C$, $\varphi(d) = D$ for some matrices $A, B, C, D \in \mathrm{H}(3,\mathbb{C})$. By Lemma 2.2, two matrices $M, N \in \mathrm{H}(3,\mathbb{C})$ commute if and only if $\vec{v}(M) \times \vec{v}(N) = 0$. Denote $\vec{v}(A) = (a_1, a_2)$ and $\vec{v}(B), \vec{v}(C), \vec{v}(D), \vec{v}(E)$ are denoted analogously. From the relations (7.2), it follows that

$$a_1c_2 = c_1a_2, \quad a_1d_2 = d_1a_2, \quad b_1c_2 = c_1b_2, \quad b_1d_2 = d_1b_2, \quad a_1b_2 \neq b_1a_2, \quad c_1d_2 \neq d_1c_2.$$

Observe first, that $a_1, a_2, b_1, b_2, c_1, c_2, d_1, d_2 \neq 0$. If, say, $a_1 = 0$, then, from the first two equations, it follows that either $a_2 = 0$ or $c_1 = d_1 = 0$. If $a_2 = 0$, then the first inequality does not hold, since $a_1b_2 = 0 = b_1a_2$, and if $c_1 = d_1 = 0$, then the second inequality does not hold, since $c_1d_2 = 0 = d_1c_2$.

Now, we can solve a_1 from the first two equalities, $\frac{a_2c_1}{c_2} = a_1 = \frac{a_2d_1}{d_2}$. That is, $c_1d_2 = d_1c_2$, which contradicts the last relation and proves our claim. □

7.3 Decidability of the identity problem in the Heisenberg group

The decidability of the identity problem in dimension three is a long standing open problem. Following our findings on non-existence of embedding into $\mathrm{SL}(3,\mathbb{Z})$, in this section we consider the decidability of an important subgroup of $\mathrm{SL}(3,\mathbb{Z})$, the Heisenberg group, which is well-known in the context of quantum mechanical systems [32, 74, 100]. Recently a few decidability results have been obtained for a knapsack variant of the membership problem in dimension three (i.e., $\mathrm{H}(3,\mathbb{Z})$), where the goal was to solve a single matrix equation with a specific order of matrices [98].

In this section, we prove that the identity problem is decidable for the Heisenberg group over rational numbers. First, we provide more intuitive solution for dimension three, i.e.,

$\mathrm{H}(3,\mathbb{Q})$, which still requires a number of techniques to estimate possible values of elements under permutations in matrix products. In the end of the section, we generalise the result for $\mathrm{H}(d,\mathbb{Q})$ case using analogies in the solution for dimension three.

Here we prove that the identity problem for matrix semigroups in the Heisenberg group over rationals is decidable by analysing the behaviour of multiplications especially in the upper-right coordinate of matrices. From Lemma 2.2, it follows that the matrix multiplication is commutative in the Heisenberg group if and only if matrices have pairwise parallel superdiagonal vectors. So we analyse two cases of products for matrices with pairwise parallel and none pairwise parallel superdiagonal vectors and then provide the algorithms that solves the problem in polynomial time. The most difficult part is to show that only limited number of conditions must be checked to guarantee the existence of a product that results in the identity.

Lemma 7.13. *Let $G = \{M_1, M_2, \ldots, M_r\} \subseteq \mathrm{H}(3,\mathbb{Q})$ be a set of matrices from the Heisenberg group such that superdiagonal vectors of matrices are pairwise parallel. If there exists a sequence of matrices $M = M_{i_1}M_{i_2}\cdots M_{i_k}$, where $i_j \in [1,r]$ for all $1 \leq j \leq k$, such that $\psi(M) = (0,0,c)$ for some $c \in \mathbb{Q}$, then,*

$$c = \sum_{j=1}^{k}(c_{i_j} - \frac{q}{2}a_{i_j}^2)$$

for some $q \in \mathbb{Q}$, dependent only on G.

Proof. Consider the sequence $M_{i_1}M_{i_2}\cdots M_{i_k}$ and let $M_i = \left(\begin{smallmatrix} 1 & a_i & c_i \\ 0 & 1 & b_i \\ 0 & 0 & 1 \end{smallmatrix}\right)$ for each $i \in [1,r]$. Since the superdiagonal vectors are parallel, we have $q = \frac{b_i}{a_i} \in \mathbb{Q}$ and thus $a_i q = b_i$ for all $i \in [1,r]$. Let us consider the product of the matrices. Then the value c is equal to

$$\begin{aligned} c &= \sum_{j=1}^{k} c_{i_j} + \sum_{\ell=1}^{k-1}\left(\sum_{j=1}^{\ell} a_{i_j}\right)a_{i_{\ell+1}}q = \sum_{j=1}^{k} c_{i_j} + \frac{1}{2}\left(\sum_{\ell=1}^{k}\sum_{j=1}^{k} a_{i_\ell}a_{i_j}q - \sum_{j=1}^{k} a_{i_j}^2 q\right) \\ &= \sum_{j=1}^{k}(c_{i_j} - \frac{q}{2}a_{i_j}^2). \end{aligned}$$

The value c would be preserved in case of reordering of matrices due to their commutativity. □

Note that the previous lemma also holds for $\mathrm{H}(3,\mathbb{R})$.

It is worth mentioning that the identity problem in the Heisenberg group is decidable if any two matrices have pairwise parallel superdiagonal vectors since now the problem reduces to solving a system of two Diophantine equations. Hence, it remains to consider the case when there exist two matrices with non-parallel superdiagonal vectors in the sequence generating the identity matrix. In the following, we prove that the identity matrix is always constructible if we can construct any matrix with the zero superdiagonal vector by using matrices with non-parallel superdiagonal vectors.

Lemma 7.14. *Let $S = \langle M_1, \ldots, M_r \rangle \subseteq \mathrm{H}(3, \mathbb{Q})$ be a finitely generated matrix semigroup. Then the identity matrix exists in S if there exists a sequence of matrices $M_{i_1} M_{i_2} \cdots M_{i_k}$, where $i_j \in [1, r]$ for all $1 \leq j \leq k$, satisfying the following properties:*

(i) $\psi(M_{i_1} M_{i_2} \cdots M_{i_k}) = (0, 0, c)$ for some $c \in \mathbb{Q}$, and

(ii) $\vec{v}(M_{i_{j_1}})$ and $\vec{v}(M_{i_{j_2}})$ are not parallel for some $j_1, j_2 \in [1, k]$.

Proof. Let $M = M_{i_1} M_{i_2} \cdots M_{i_k}$ and $\psi(M) = (0, 0, c)$ for some $c \in \mathbb{Q}$. If $c = 0$, then M is the identity matrix, hence we assume that $c > 0$ as the case of $c < 0$ is symmetric.

Given that M_i is the ith generator and $\psi(M_i) = (a_i, b_i, c_i)$, we have $\sum_{j=1}^{k} a_{i_j} = 0$ and $\sum_{j=1}^{k} b_{i_j} = 0$. Since $c > 0$, the following also holds:

$$c = \sum_{\ell=1}^{k-1} \sum_{j=1}^{\ell} a_{i_j} b_{i_{\ell+1}} + \sum_{j=1}^{k} c_{i_j} > 0. \tag{7.8}$$

If the matrix semigroup $S \subseteq \mathrm{H}(3, \mathbb{Q})$ has two different matrices M_1 and M_2 such that $\psi(M_1) = (0, 0, c_1)$ and $\psi(M_2) = (0, 0, c_2)$ and $c_1 c_2 < 0$, then the identity matrix exists in S. Let $\psi(M_1) = (0, 0, \frac{p_1}{q_1})$ and $\psi(M_2) = (0, 0, \frac{p_2}{q_2})$, where $p_1, q_1, q_2 \in \mathbb{Z}$ are positive and $p_2 \in \mathbb{Z}$ is negative. Then it is easy to see that the matrix $M_1^{-q_1 p_2} M_2^{q_2 p_1}$ exists in S and that $\psi(M_1^{-q_1 p_2} M_2^{q_2 p_1}) = (0, 0, 0)$.

Now we will prove that if S contains a matrix M such that $\psi(M) = (0, 0, c)$, where $c > 0$, then there also exists a matrix M' such that $\psi(M') = (0, 0, c')$, where $c' < 0$.

First, we classify the matrices into four types as follows. A matrix with a superdiagonal vector (a, b) is classified as

1) the $(+,+)$-type if $a, b > 0$,

2) the $(+,-)$-type if $a \geq 0$ and $b \leq 0$,

3) the $(-,-)$-type if $a, b < 0$, and

4) the $(-,+)$-type if $a < 0$ and $b > 0$.

Let $G = \{M_1, M_2, \ldots, M_r\}$ be the generating set of the matrix semigroup S. Then $G = G_{(+,+)} \sqcup G_{(+,-)} \sqcup G_{(-,-)} \sqcup G_{(-,+)}$ such that $G_{(\xi_1,\xi_2)}$ is the set of matrices of the (ξ_1, ξ_2)-type, where $\xi_1, \xi_2 \in \{+, -\}$.

Recall that we assume $M = M_{i_1} \cdots M_{i_k}$ and $\psi(M) = (0, 0, c)$ for some $c > 0$. The main idea of the proof is to generate a matrix M' such that $\psi(M') = (0, 0, c')$ for some $c' < 0$ by duplicating the matrices in the sequence $M = M_{i_1} \cdots M_{i_k}$ multiple times and reshuffling. Note that any permutation of the sequence generating the matrix M such that $\psi(M) = (0, 0, c)$ still generates matrices M' such that $\psi(M') = (0, 0, c')$ since the multiplication of matrices exchanges the first two coordinates in a commutative way. Moreover, we can still obtain matrices M'' such that $\psi(M'') = (0, 0, c'')$ for some $c'' \in \mathbb{Q}$ if we shuffle two different permutations of the sequence $M_{i_1} \cdots M_{i_k}$ by the same reason.

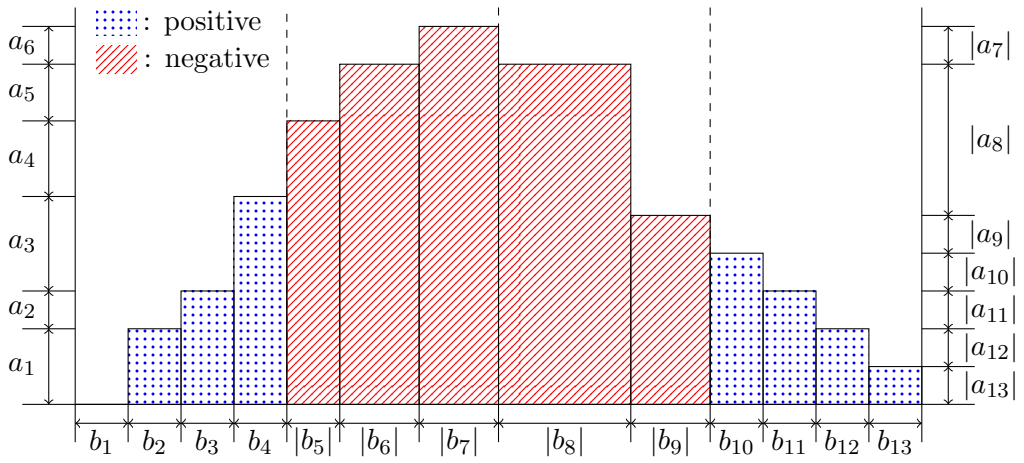


Figure 7.4: The histogram describes how the upper-right corner of $M_1 \cdots M_{13}$ is computed by multiplications. The blue dotted (red lined) area implies the value which will be added to (subtracted from) the upper-right corner of the final matrix after multiplications of matrices in the sequence.

Let us illustrate the idea with the following example. See Figure 7.4 and Figure 7.5 for pictorial descriptions of the idea. Let $\{M_i \mid 1 \leq i \leq 4\} \subseteq G_{(+,+)}$, $\{M_i \mid 5 \leq i \leq 7\} \subseteq G_{(+,-)}$, $\{M_i \mid 8 \leq i \leq 9\} \subseteq G_{(-,-)}$, and $\{M_i \mid 10 \leq i \leq 13\} \subseteq G_{(-,+)}$. Then assume that $M_1 M_2 \cdots M_{13} = \left(\begin{smallmatrix} 1 & 0 & x \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{smallmatrix}\right)$, where x is computed by equation (7.8). As we mentioned above,

x changes if we change the order of multiplications. In this example, we first multiply $(+,+)$-type matrices and accumulate the values in the superdiagonal coordinates since these matrices have positive values in the coordinates. Indeed, the blue dotted area implies the value we add to the upper-right corner by multiplying such matrices. Then we multiply $(+,-)$-type matrices and still increase the 'a'-value. The 'b'-values in $(+,-)$-type matrices are negative thus, the red lined area is subtracted from the upper-right corner. We still subtract by multiplying $(-,-)$-type matrices since the accumulated 'a'-value is still positive and 'b'-values are negative. Then we finish the multiplication by adding exactly the last blue dotted area to the upper-right corner. It is easy to see that the total subtracted value is larger than the total added value.

However, we cannot guarantee that x is negative since $\sum_{i=1}^{13} c_i$ could be larger than the contribution from the superdiagonal coordinates. This is why we need to copy the sequence of matrices generating the matrix corresponding to the triple $(0,0,c)$ for some $c \in \mathbb{Q}$. In Figure 7.5, we describe an example where we duplicate the sequence eight times and shuffle and permute them in order to minimise the value in the upper-right corner. Now the lengths of both axes are m ($m = 8$ in this example) times larger than before and it follows that the area also grows quadratically in m. Since the summation $m \cdot \sum_{i=1}^{13} c_i$ grows linearly in m, we have $x < 0$ when m is large enough.

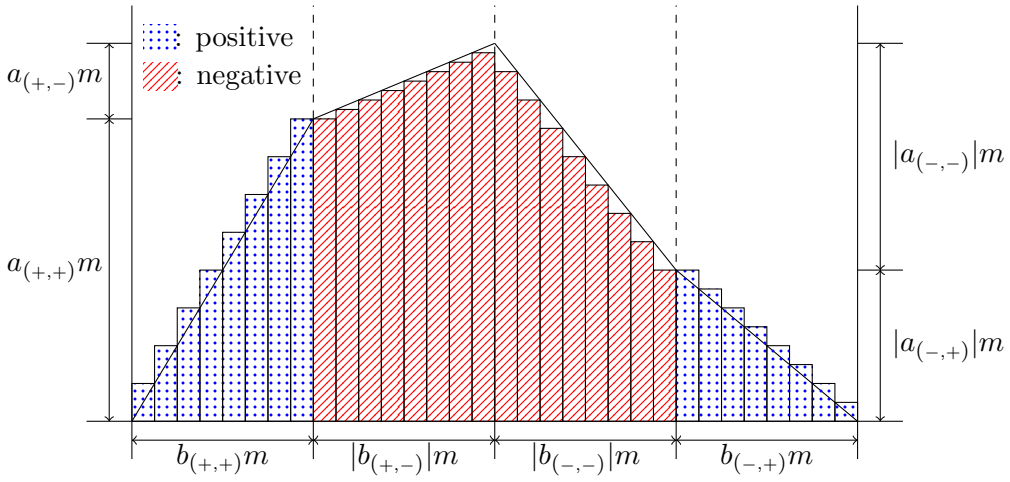


Figure 7.5: The histogram describes how the value in the upper-right corner of matrix $M^m_{(+,+)}M^m_{(+,-)}M^m_{(-,-)}M^m_{(-,+)}$ is computed by multiplications. Here $m = 8$.

For each $\xi_1, \xi_2 \in \{+,-\}$, let us define multisets $S_{(\xi_1,\xi_2)}$ that are obtained from the

sequence $M_{i_1} \cdots M_{i_k}$ by partitioning the product according to the matrix types. That is, $S_{(\xi_1,\xi_2)}$ contains exactly the matrices of (ξ_1, ξ_2)-type in the product (possibly with several copies of each matrix).

For each $\xi_1, \xi_2 \in \{+, -\}$, let us define $a_{(\xi_1,\xi_2)}, b_{(\xi_1,\xi_2)}, c_{(\xi_1,\xi_2)}$ such that

$$(a_{(\xi_1,\xi_2)}, b_{(\xi_1,\xi_2)}, c_{(\xi_1,\xi_2)}) = \sum_{M \in S_{(\xi_1,\xi_2)}} \psi(M).$$

In other words, $a_{(\xi_1,\xi_2)}$ ($b_{(\xi_1,\xi_2)}$ and $c_{(\xi_1,\xi_2)}$, respectively) is the sum of the values in the 'a' ('b' and 'c', respectively) coordinate from the matrices in the multiset $S_{(\xi_1,\xi_2)}$.

Now consider a permutation of the sequence $M_{i_1} \cdots M_{i_k}$, where the first part of the sequence only consists of the (+,+)-type matrices, the second part only consists of the (+,−)-type matrices, the third part only consists of the (−,−)-type, and finally the last part only consists of the (−,+)-type.

Let us denote by $M_{(+,+)}$ the matrix which results from the multiplication of the first part, namely, $M_{(+,+)} = \prod_{M \in S_{(+,+)}} M$. Then $\psi(M_{(+,+)}) = (a_{(+,+)}, b_{(+,+)}, x_{(+,+)})$ holds, where $x_{(+,+)} < c_{(+,+)} + a_{(+,+)} b_{(+,+)}$. Let us define $M_{(+,-)}$, $M_{(-,-)}$ and $M_{(-,+)}$ in a similar fashion. Note that for $M_{(+,-)}$ and $M_{(-,+)}$, the term x is bounded from below.

Now we claim that there exists an integer $m > 0$ such that $M^m_{(+,+)} M^m_{(+,-)} M^m_{(-,-)} M^m_{(-,+)}$ corresponds to the triple $(0, 0, c')$ for some $c' < 0$. Let N be a matrix in $\mathrm{H}(3, \mathbb{Q})$ and $\psi(N) = (a, b, c)$. Then the upper-triangular coordinates of the mth power of N are calculated as follows: $\psi(N^m) = (am, bm, cm + ab \cdot \frac{1}{2} m(m-1))$.

$$\begin{aligned}
z_1 &= |a_{(+,+)}||b_{(+,+)}| \cdot \frac{1}{2} m(m-1), \\
z_2 &= m^2 |a_{(+,+)}||b_{(+,-)}| + |a_{(+,-)}||b_{(+,-)}| \cdot \frac{1}{2} m(m-1), \\
z_3 &= |a_{(-,+)}||b_{(-,-)}| m^2 + |a_{(-,-)}||b_{(-,-)}| \cdot \frac{1}{2} m(m-1) \text{ and} \\
z_4 &= |a_{(-,+)}||b_{(-,+)}| \cdot \frac{1}{2} m(m-1).
\end{aligned}$$

Table 7.2: Values z_1, z_2, z_3 and z_4 in the product $M^m_{(+,+)} M^m_{(+,-)} M^m_{(-,-)} M^m_{(-,+)}$.

Next, we consider how the upper-triangular coordinates are affected by multiplication of matrices $M^m_{(+,+)}$, $M^m_{(+,-)}$, $M^m_{(-,-)}$ and $M^m_{(-,+)}$. Let us consider the first part of the product,

$M^m_{(+,+)}$, that is, $\psi(M^m_{(+,+)}) = (a_{(+,+)}m, b_{(+,+)}m, x_{(+,+)}m + z_1)$, where z_1 can be found in Table 7.2. Now we multiply $M^m_{(+,+)}$ by the second part $M^m_{(+,-)}$. Then the resulting matrix $M^m_{(+,+)}M^m_{(+,-)}$ corresponds to

$$\psi(M^m_{(+,+)}M^m_{(+,-)}) = ((a_{(+,+)} + a_{(+,-)})m, (b_{(+,+)} + b_{(+,-)})m, (x_{(+,+)} + x_{(+,-)})m + z_1 - z_2),$$

where z_2 can be found in Table 7.2. Similarly, we compute z_3 and z_4 that will be added to the upper-right corner as a result of multiplying $M^m_{(-,-)}$ and $M^m_{(-,+)}$ and present them in Table 7.2.

After the multiplying all four parts, we have

$$\psi(M^m_{(+,+)}M^m_{(+,-)}M^m_{(-,-)}M^m_{(-,+)}) = (0, 0, (x_{(+,+)} + x_{(+,-)} + x_{(-,-)} + x_{(-,+)})m + z_1 - z_2 - z_3 + z_4).$$

Denote $z = z_1 - z_2 - z_3 + z_4$. From Table 7.2, we can see that z can be represented as a quadratic equation of m and that the coefficient of m^2 is always negative if $S_{(\xi_1,\xi_2)} \neq \emptyset$ for each $\xi_1, \xi_2 \in \{+, -\}$. That is, the coefficient of m^2 is

$$\frac{1}{2}(|a_{(+,+)}||b_{(+,+)}| + |a_{(-,+)}||b_{(-,+)}|) - \frac{1}{2}(|a_{(+,-)}||b_{(+,-)}| + |a_{(-,-)}||b_{(-,-)}|) \\ + |a_{(+,+)}||b_{(+,-)}| + |a_{(-,+)}||b_{(-,-)}|.$$

Let us simplify the equation by denoting $|a_{(+,+)}| + |a_{(+,-)}| = |a_{(-,+)}| + |a_{(-,-)}| = a'$ and $|b_{(+,+)}| + |b_{(-,+)}| = |b_{(+,-)}| + |b_{(-,-)}| = b'$. Note, that the equations hold as we are considering the product, where 'a' and 'b' elements add up to zero. Then

$$\begin{aligned} a'b' &= a'(|b_{(+,-)}| + |b_{(-,-)}|) = a'|b_{(+,-)}| + a'|b_{(-,-)}| \\ &= (|a_{(+,+)}| + |a_{(+,-)}|)|b_{(+,-)}| + (|a_{(-,+)}| + |a_{(-,-)}|)|b_{(-,-)}|. \end{aligned}$$

Now the coefficient of m^2 in z can be written as

$$-a'b' + \frac{1}{2}(|a_{(+,+)}||b_{(+,+)}| + |a_{(-,+)}||b_{(-,+)}| + |a_{(-,-)}||b_{(-,-)}| + |a_{(+,-)}||b_{(+,-)}|). \tag{7.9}$$

Without loss of generality, suppose that $|a_{(+,+)}| \geq |a_{(-,+)}|$. Then we have

$$|a_{(+,+)}||b_{(+,+)}| + |a_{(-,+)}||b_{(-,+)}| \leq |a_{(+,+)}|b' \text{ and } |a_{(-,-)}||b_{(-,-)}| + |a_{(+,-)}||b_{(+,-)}| \leq |a_{(-,-)}|b'.$$

From $(|a_{(+,+)}|+|a_{(-,-)}|)b' \leq 2a'b'$, we can see that the coefficient of the highest power of the variable is negative in z if $|a_{(+,+)}|+|a_{(-,-)}| < 2a'$. By comparing two terms in equation (7.9), we can see that the coefficient is negative if all subsets $S_{(-,+)}$, $S_{(+,-)}$, $S_{(+,+)}$ and $S_{(-,-)}$ are non-empty. Since the coefficient of the highest power of the variable is negative, z becomes negative when m is large enough. Therefore, we have a matrix corresponding to the triple $(0,0,c')$ for some $c' < 0$ as a product of multiplying matrices in the generating set and the identity matrix is also reachable.

It should be noted that there are some subcases where some of subsets from $S_{(+,+)}$, $S_{(-,+)}$, $S_{(+,-)}$, and $S_{(-,-)}$ are empty. We examine all possible cases and prove that the coefficient of m^2 is negative in every case and the matrix with a negative number in the corner is constructible. First, we prove that the coefficient of m^2 in z is negative when only one of the subsets from $S_{(+,+)}$. $S_{(+,-)}$, $S_{(-,-)}$, and $S_{(-,+)}$ is empty as follows:

Assume that only $S_{(+,+)} = \emptyset$. In this case, note that $|a_{(+,-)}| = a'$ and $|b_{(-,+)}| = b'$ since $|a_{(+,+)}| = |b_{(+,+)}| = 0$ by $S_{(+,+)} = \emptyset$ being empty. Then the coefficient of m^2 becomes

$$-a'b' + \frac{|a_{(-,+)}|b' + |a_{(-,-)}||b_{(-,-)}| + a'|b_{(+,-)}|}{2}.$$

We can see that the coefficient can be at most 0 since $|a_{(-,+)}|b'$ and $|a_{(-,-)}||b_{(-,-)}|+a'|b_{(+,-)}|$ can be maximised to $a'b'$. If we maximise $|a_{(-,+)}|b'$ by setting $|a_{(-,+)}| = a'$, then $|a_{(-,-)}|$ is 0 since $|a_{(+,+)}| + |a_{(-,+)}| = a'$. Then $|a_{(-,-)}||b_{(-,-)}| + a'|b_{(+,-)}|$ can be $a'b'$ only when $|b_{(+,-)}| = b'$. This leads to the set $S_{(-,-)}$ being empty since we have $|a_{(-,-)}| = 0$ and $|b_{(-,-)}| = 0$. Therefore, we have a contradiction.

The remaining cases, $S_{(+,-)} = \emptyset$, or $S_{(-,-)} = \emptyset$, or $S_{(-,+)} = \emptyset$ are proven analogously.

Figure 7.6 shows the cases when one of subsets from $S_{(+,+)}$, $S_{(-,+)}$, $S_{(+,-)}$, and $S_{(-,-)}$ is empty. Lastly, it remains to consider the cases where two of the subsets are empty. Note that we do not consider the cases where three of the subsets are empty because the sum of a's and b's cannot be both zero in such cases. Here we assume one of $S_{(+,+)}$ and $S_{(-,-)}$ contains two matrices whose superdiagonal vectors are not parallel by the statement of this lemma. Then we can always make the negative contribution larger by using matrices with different superdiagonal vectors. See Figure 7.7 for an example. More formally, we consider the two cases as follows:

Assume first that $S_{(+,+)} = \emptyset$ and $S_{(-,-)} = \emptyset$. Without loss of generality, assume that $S_{(-,+)}$ contains two matrices M_1 and M_2 with non-parallel superdiagonal vectors. Let $\vec{v}(M_1) = (a_1, b_1)$ and $\vec{v}(M_2) = (a_2, b_2)$ be superdiagonal vectors for M_1 and M_2,

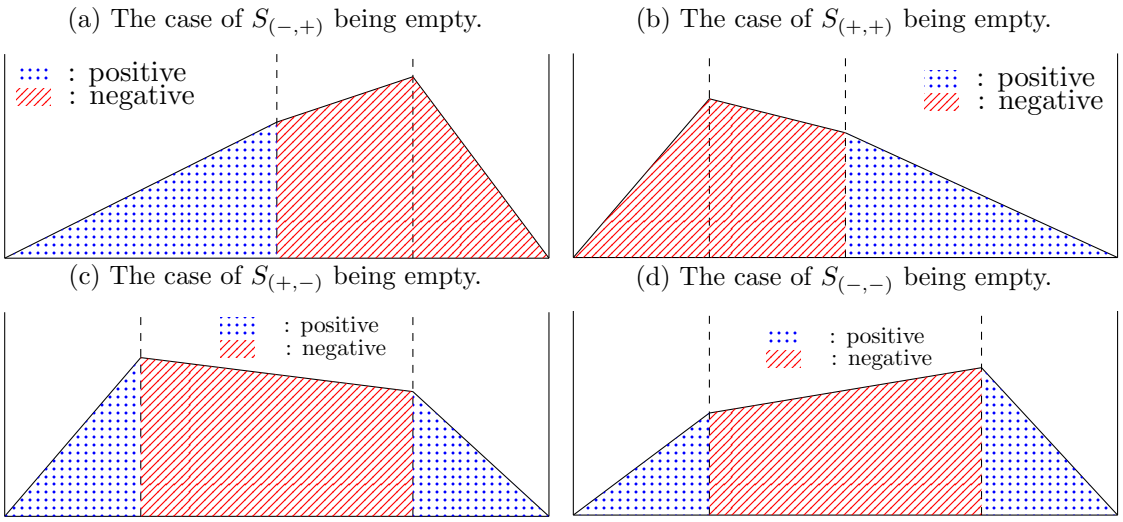


Figure 7.6: Subcases where one of the subsets from $S_{(+,+)}$, $S_{(-,+)}$, $S_{(+,-)}$, and $S_{(-,-)}$ is empty.

respectively, such that $|\frac{a_1}{b_1}| > |\frac{a_2}{b_2}|$. To simplify the proof, we assume the set $S_{(+,-)}$ only uses one matrix M_3, where $\vec{v}(M_3) = (a_3, b_3)$, to generate a matrix with a zero superdiagonal vector. This implies that $a_1x + a_2y + a_3 = 0$ and $b_1x + b_2y + b_3 = 0$ for some $x, y \in \mathbb{Q}$. Here the idea is that we first multiply the matrix M_1 and then multiply M_2 later. For instance, we first multiply M_1^m and then M_2^m. Then the coefficient of the highest power in z becomes $\frac{-a'b'+2|a_2||b_1|+|a_1||b_1|+|a_2||b_2|}{2}$. Since $a' = |a_1| + |a_2|$ and $b' = |b_1| + |b_2|$, the coefficient of m^2 is now $\frac{|a_2||b_1|-|a_1||b_2|}{2}$. By the supposition $|\frac{a_1}{b_1}| > |\frac{a_2}{b_2}|$, we prove that the coefficient of the highest power in z is always negative.

The second case, where $S_{(+,-)} = \emptyset$ and $S_{(-,+)} = \emptyset$, is proven analogously.

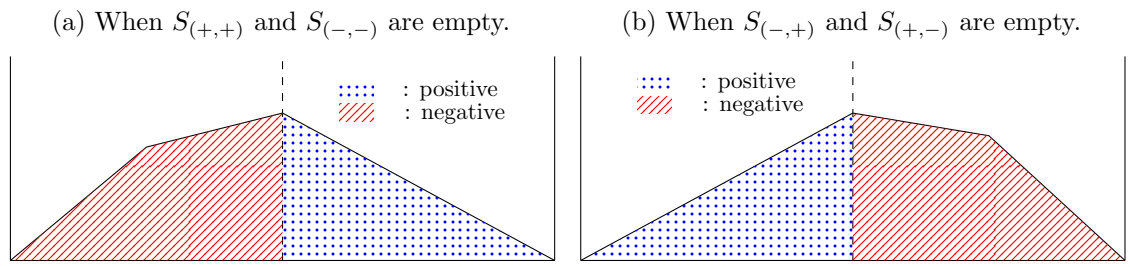


Figure 7.7: Subcases where two of the subsets from $S_{(+,+)}$, $S_{(-,+)}$, $S_{(+,-)}$, and $S_{(-,-)}$ are empty.

As we have proven that it is always possible to construct a matrix M' such that $\psi(M') = (0, 0, c')$ for some $c' < 0$, we complete the proof. □

Note that in the above proof, we do not give optimal bounds on number of repetitions of a sequence.

We illustrate Lemma 7.14 in the next example.

Example 7.15. Consider a semigroup S generated by matrices

$$\begin{pmatrix} 1 & -4 & 20 \\ 0 & 1 & -6 \\ 0 & 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 3 & 20 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & -1 & 20 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 2 & 20 \\ 0 & 1 & 7 \\ 0 & 0 & 1 \end{pmatrix}.$$

A simple calculation shows that a product of the four matrices (in any order) is a matrix M such that $\psi(M) = (0, 0, 80 + x)$ for some $x \in \mathbb{Z}$. Our goal, is to minimise x by multiplying the matrices in a different order. Denote the given matrices by $M_{(+,+)} = \left(\begin{smallmatrix} 1 & 2 & 20 \\ 0 & 1 & 7 \\ 0 & 0 & 1 \end{smallmatrix}\right)$, $M_{(+,-)} = \left(\begin{smallmatrix} 1 & 3 & 20 \\ 0 & 1 & -2 \\ 0 & 0 & 1 \end{smallmatrix}\right)$, $M_{(-,-)} = \left(\begin{smallmatrix} 1 & -4 & 20 \\ 0 & 1 & -6 \\ 0 & 0 & 1 \end{smallmatrix}\right)$ and $M_{(-,+)} = \left(\begin{smallmatrix} 1 & -1 & 20 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{smallmatrix}\right)$, and

$$N_1 = M_{(+,+)}M_{(+,-)}M_{(-,-)}M_{(-,+)} = \begin{pmatrix} 1 & 0 & 47 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

That is, $x = -33$. By considering several copies of the product, we can have a negative value in the top right corner. Indeed, consider the product of 16 matrices

$$N_2 = M^4_{(+,+)}M^4_{(+,-)}M^4_{(-,-)}M^4_{(-,+)} = \begin{pmatrix} 1 & 0 & -22 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Since, we have a matrix with negative value in the top corner, the identity matrix can be generated for example by the product $N_1^{22}N_2^{47}$.

Theorem 7.16. *The identity problem for finitely generated matrix semigroups in the Heisenberg group* $\mathrm{H}(3, \mathbb{Q})$ *is decidable in polynomial time.*

Proof. Let S be the matrix semigroup in $\mathrm{H}(3, \mathbb{Q})$ generated by the set $G = \{M_1, \ldots, M_r\}$. There are two possible cases of having the identity matrix in the matrix semigroup in $\mathrm{H}(3, \mathbb{Q})$.

Either the identity matrix is generated by a product of matrices with pairwise parallel superdiagonal vectors or there are at least two matrices with non-parallel superdiagonal vectors.

Consider the first case. Lemma 7.13 provides a formula to compute the value in the top corner regardless of the order of the multiplications. That is, we need to solve a system of linear Diophantine equations with solutions over non-negative integers. We partition the set G into several disjoint subsets $G_1, G_2, \ldots, G_s$, where s is at most r, and each subset contains matrices with parallel superdiagonal vectors. Since superdiagonal vectors being parallel is a transitive and symmetric property, each matrix needs to be compared to a representative of each subset. If there are no matrices with parallel superdiagonal vectors, then there are r subsets G_i containing exactly one matrix and $O(r^2)$ tests were done. Let us consider $G_i = \{M_{k_1}, \ldots, M_{k_{s_i}}\}$, i.e., one of the subsets containing s_i matrices and $\psi(M_{k_j}) = (a_{k_j}, b_{k_j}, c_{k_j})$. By considering each matrix as the trivial product, by Lemma 7.13, we write

$$M_{k_j} = \begin{pmatrix} 1 & a_{k_j} & c_{k_j} - \frac{q_i}{2} a_{k_j}^2 \\ 0 & 1 & a_{k_j} q_i \\ 0 & 0 & 1 \end{pmatrix},$$

for a fixed $q_i \in \mathbb{Q}$.

We solve the system of two linear Diophantine equations $A\mathbf{y} = \mathbf{0}$, where

$$A = \begin{pmatrix} a_{k_1} & a_{k_2} & \cdots & a_{k_{s_i}} \\ 2c_{k_1} - q_i a_{k_1}^2 & 2c_{k_2} - q_i a_{k_2}^2 & \cdots & 2c_{k_{s_i}} - q_i a_{k_{s_i}}^2 \end{pmatrix}$$

and $\mathbf{y} \geq 0$. The first row corresponds to element a being zero, and thus also the superdiagonal vector being zero, and the second row to the upper corner being zero.

It is obvious that the identity matrix is in the semigroup if we have a solution in the system of homogeneous linear Diophantine equations for any subset G_i. That is, we need to solve at most r systems of homogeneous linear Diophantine equations.

Next, we consider the second case, where by Lemma 7.14, it is enough to check whether there exists a sequence of matrices generating a matrix with zero superdiagonal vector and containing two matrices with non-parallel superdiagonal vectors. Let us say that $M_{i_1}, M_{i_2} \in G$, where $1 \leq i_1, i_2 \leq r$ are the two matrices. Recall that $G = \{M_1, M_2, \ldots, M_r\}$ is a generating set of the matrix semigroup and let $\psi(M_i) = (a_i, b_i, c_i)$ for all $1 \leq i \leq r$. We

can see that there exists such a product containing the two matrices by solving a system of two homogeneous linear Diophantine equations of the form $B\mathbf{y} = \mathbf{0}$, where

$$B = \begin{pmatrix} a_1 & a_2 & \cdots & a_r \\ b_1 & b_2 & \cdots & b_r \end{pmatrix},$$

with an additional constraint that the numbers in the solution $\mathbf{y}$ that correspond to M_{i_1} and M_{i_2} are non-zero since we must use these two matrices in the product. We repeat this process at most $r(r-1)$ times until we find a solution. Therefore, the problem reduces again to solving at most $O(r^2)$ systems of linear Diophantine equations.

Finally, we conclude the proof by mentioning that the identity problem for matrix semigroups in the Heisenberg group over rationals can be decided in polynomial time as a system of two homogeneous linear Diophantine equations can be solved in polynomial time when the solution is restricted to non-negative integers [140]. □

Next, we generalise the above algorithm for the identity problem in the Heisenberg group $\mathrm{H}(3, \mathbb{Q})$ to the domain of the Heisenberg groups for any dimension over the rational numbers. Similarly to the case of dimension three, we establish the following result for the case of matrices where multiplication is commutative.

Lemma 7.17. *Let $G = \{M_1, M_2, \ldots, M_r\} \subseteq \mathrm{H}(d, \mathbb{Q})$ be a set of matrices from the Heisenberg group such that $\psi(M_i) = (\mathbf{a}_i, \mathbf{b}_i, c)$ and $\psi(M_j) = (\mathbf{a}_j, \mathbf{b}_j, c)$ and $\mathbf{a}_i \cdot \mathbf{b}_j = \mathbf{a}_j \cdot \mathbf{b}_i$ for any $1 \leq i \neq j \leq r$. If there exists a sequence of matrices $M = M_{i_1} M_{i_2} \cdots M_{i_k}$, where $i_j \in [1, r]$ for all $1 \leq j \leq k$, such that $\psi(M) = (\mathbf{0}, \mathbf{0}, c)$ for some $c \in \mathbb{Q}$, then,*

$$c = \sum_{j=1}^{k} (c_{i_j} - \frac{1}{2} \mathbf{a}_{i_j} \cdot \mathbf{b}_{i_j}).$$

Proof. Consider the sequence $M_{i_1} M_{i_2} \cdots M_{i_k}$ and let $\psi(M_i) = (\mathbf{a}_i, \mathbf{b}_i, c_i)$ for each $i \in [1, r]$. From the multiplication of matrices, we have the following equation:

$$\begin{aligned} c &= \sum_{j=1}^{k} c_{i_j} + \sum_{\ell=1}^{k-1} \left(\sum_{j=1}^{\ell} \mathbf{a}_{i_j} \right) \cdot \mathbf{b}_{i_{\ell+1}} = \sum_{j=1}^{k} c_{i_j} + \frac{1}{2} \left(\sum_{\ell=1}^{k} \sum_{j=1}^{k} \mathbf{a}_{i_\ell} \cdot \mathbf{b}_{i_j} - \sum_{j=1}^{k} \mathbf{a}_{i_j} \cdot \mathbf{b}_{i_j} \right) \\ &= \sum_{j=1}^{k} (c_{i_j} - \frac{1}{2} \mathbf{a}_{i_j} \cdot \mathbf{b}_{i_j}). \end{aligned}$$

From the above equation, we prove the statement claimed in the lemma. Moreover, due to the commutativity of multiplication, the value c does not change even if we change the order of multiplicands. □

Lemma 7.14 does not generalise to $\mathrm{H}(d, \mathbb{Q})$ in the same way as we cannot classify matrices according to types to control the value in upper-right corner, so we use a different technique to prove that the value in the upper corner will be diverging to both positive and negative infinity quadratically as we repeat the same sequence generating any matrix M such that $\psi(M) = (\mathbf{0}, \mathbf{0}, c)$.

Lemma 7.18. *Let $S = \langle M_1, \ldots, M_r \rangle \subseteq \mathrm{H}(d, \mathbb{Q})$ be a finitely generated matrix semigroup. Then the identity matrix exists in S if there exists a sequence of matrices $M_{i_1} M_{i_2} \cdots M_{i_k}$, where $i_j \in [1, r]$ for all $1 \leq j \leq k$, satisfying the following properties:*

(i) $\psi(M_{i_1} M_{i_2} \cdots M_{i_k}) = (\mathbf{0}, \mathbf{0}, c)$ for some $c \in \mathbb{Q}$, and

(ii) $\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} \neq \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_1}}$ for some $j_1, j_2 \in [1, k]$, where $\psi(M_i) = (\mathbf{a}_i, \mathbf{b}_i, c_i)$ for $1 \leq i \leq r$.

Proof. From the first property claimed in the lemma, we know that any permutation of the sequence of matrix multiplications of $M_{i_1} \cdots M_{i_k}$ results in matrices M' such that $\psi(M') = (\mathbf{0}, \mathbf{0}, y)$ for some $y \in \mathbb{Q}$ since the multiplication of matrices in the Heisenberg group performs additions of vectors which is commutative in the top row and the rightmost column excluding the upper-right corner. From the commutative behaviour in the horizontal and vertical vectors of matrices in the Heisenberg group, we also know that if we duplicate the matrices in the sequence $M_{i_1} \cdots M_{i_k}$ and multiply the matrices in any order, then the resulting matrix has a non-zero coordinate in the upper triangular coordinates only in the upper right corner.

Now let $j_1, j_2 \in [1, k]$ be two indices such that $\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} \neq \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_1}}$ as claimed in the lemma. Then consider the following matrix M_d that can be obtained by duplicating the sequence $M_{i_1} \cdots M_{i_k}$ of matrices into ℓ copies and shuffle the order as follows: $M_d = M_{i_{j_1}}^{\ell} M_{i_{j_2}}^{\ell} M_x^{\ell}$, where M_x is a matrix that is obtained by multiplying the matrices in $M_{i_1} \cdots M_{i_k}$ except the two matrices M_{j_1} and M_{j_2}. Then it is clear that $\psi(M_d) = (\mathbf{0}, \mathbf{0}, z)$ for some z. Let us say that $\psi(M_x) = (\mathbf{a}_x, \mathbf{b}_x, c_x)$. Then it is easy to see that $\mathbf{a}_{i_{j_1}} + \mathbf{a}_{i_{j_2}} + \mathbf{a}_x = \mathbf{0}$ and $\mathbf{b}_{i_{j_1}} + \mathbf{b}_{i_{j_2}} + \mathbf{b}_x = \mathbf{0}$. Now we show that we can always construct two matrices that have only one non-zero rational number in the upper right corner with different signs.

First, let us consider the ℓth power of the matrix $M_{i_{j_1}}$ as follows:

$$\psi(M_{i_{j_1}}^{\ell}) = (\mathbf{a}_{i_{j_1}}\ell, \mathbf{b}_{i_{j_1}}\ell, c_{i_{j_1}}\ell + \sum_{h=1}^{\ell-1} h(\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_1}})) = (\mathbf{a}_{i_{j_1}}\ell, \mathbf{b}_{i_{j_1}}\ell, c_{i_{j_1}}\ell + \mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_1}} \frac{(\ell-1)\ell}{2}).$$

It follows that the matrix M_d satisfies the equation $\psi(M_d) = (\mathbf{0}, \mathbf{0}, z)$ such that

$$\begin{aligned} z &= y\ell + (\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_1}} + \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_2}} + \mathbf{a}_x \cdot \mathbf{b}_x)\frac{(\ell-1)\ell}{2} + (\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} + (\mathbf{a}_{i_{j_1}} + \mathbf{a}_{i_{j_2}}) \cdot \mathbf{b}_x)\ell^2 \\ &= \frac{1}{2}((\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_1}} + \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_2}} + \mathbf{a}_x \cdot \mathbf{b}_x) + 2(\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} + (\mathbf{a}_{i_{j_1}} + \mathbf{a}_{i_{j_2}}) \cdot \mathbf{b}_x))\ell^2 \\ &\quad + \frac{1}{2}(2y - (\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_1}} + \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_2}} + \mathbf{a}_x \cdot \mathbf{b}_x))\ell. \end{aligned}$$

Now the coefficient of the highest term ℓ^2 in z can be simplified as follows:

$$\begin{aligned} &\frac{1}{2}((\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_1}} + \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_2}} + \mathbf{a}_x \cdot \mathbf{b}_x) + 2(\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} + (\mathbf{a}_{i_{j_1}} + \mathbf{a}_{i_{j_2}}) \cdot \mathbf{b}_x)) \\ &\quad = \frac{1}{2}((\mathbf{a}_{i_{j_1}} + \mathbf{a}_{i_{j_2}}) \cdot (\mathbf{b}_{i_{j_1}} + \mathbf{b}_{i_{j_1}}) + \mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} - \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_1}} + (\mathbf{a}_{i_{j_1}} + \mathbf{a}_{i_{j_2}}) \cdot \mathbf{b}_x) \\ &\quad = \frac{1}{2}((-\mathbf{a}_x) \cdot (-\mathbf{b}_x) + \mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} - \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_1}} + (-\mathbf{a}_x) \cdot \mathbf{b}_x) \\ &\quad = \frac{1}{2}(\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} - \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_1}}). \end{aligned}$$

By the second property claimed in the lemma, we know that the coefficient of the highest term ℓ^2 in z cannot be zero. Moreover, the value of z will be diverging to negative or positive infinity depending on the sign of $\mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}} - \mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_1}}$. Now we consider a different matrix M_e which is defined to be the following product $M_{i_{j_2}}^{\ell} M_{i_{j_1}}^{\ell} M_x^{\ell}$ and say that $\psi(M_e) = (\mathbf{0}, \mathbf{0}, e)$ for some $e \in \mathbb{Q}$. Since we have changed the role of two matrices $M_{i_{j_1}}$ and $M_{i_{j_2}}$, the value of e can be represented by a quadratic equation where the coefficient of the highest term is $\mathbf{a}_{i_{j_2}} \cdot \mathbf{b}_{i_{j_1}} - \mathbf{a}_{i_{j_1}} \cdot \mathbf{b}_{i_{j_2}}$. Therefore, we have proved that it is always possible to construct two matrices that have only one non-zero rational number in the upper right corner with different signs. Then, as in the proof Lemma 7.14, the identity matrix always exists in the semigroup as we can multiply these two matrices correct number of times to have zero in the upper right coordinate as well. □

Next, we prove that the identity problem is decidable for d-dimensional Heisenberg matrices. In contrast to Theorem 7.16, we do not claim that the problem is decidable

in polynomial time since one of the steps of the proof is to partition matrices according to dot products, which cannot be extended to higher dimensions than three. For higher dimensions, partitioning matrices according to dot products takes an exponential time in the number of matrices in the generating set.

Theorem 7.19. *The identity problem for finitely generated matrix semigroups in the Heisenberg group* $\mathrm{H}(d, \mathbb{Q})$ *is decidable.*

Proof. Similarly to the proof of Theorem 7.16, there are two ways the identity matrix can be generated. Either all the matrices commute or there are at least two matrices that do not commute.

Let S be the matrix semigroup in $\mathrm{H}(d, \mathbb{Q})$ generated by the set $G = \{M_1, M_2, \ldots, M_r\}$. Consider matrices N_1, N_2 and N_3, such that $\psi(N_1) = (\mathbf{a}_1, \mathbf{b}_1, c_1)$, $\psi(N_2) = (\mathbf{a}_2, \mathbf{b}_2, c_2)$ and $\psi(N_3) = (\mathbf{a}_3, \mathbf{b}_3, c_3)$. If $\mathbf{a}_1 \cdot \mathbf{b}_2 = \mathbf{a}_2 \cdot \mathbf{b}_1$ and $\mathbf{a}_2 \cdot \mathbf{b}_3 = \mathbf{a}_3 \cdot \mathbf{b}_2$, it does not imply that $\mathbf{a}_1 \cdot \mathbf{b}_3 = \mathbf{a}_3 \cdot \mathbf{b}_1$. Therefore, the number of subsets of G, where each subset contains matrices that commute with other matrices in the same subset, is exponential in r as two different subsets are not necessarily disjoint.

Now we examine whether it is possible to generate the identity matrix by multiplying matrices in each subset by Lemma 7.17. If it is not possible, we need to consider the case of having two matrices that do not commute with each other in the product with zero values in the upper-triangular coordinates except the corner. Let us say that $M_{i_1}, M_{i_2} \in G$, where $1 \leq i_1, i_2 \leq r$ are the two matrices. Recall that $G = \{M_1, M_2, \ldots, M_r\}$ is a generating set of the matrix semigroup and let $\psi(M_i) = (\mathbf{a}_i, \mathbf{b}_i, c_i)$ for all $1 \leq i \leq r$. We also denote the mth element of the vector $\mathbf{a}_i$ (respectively, $\mathbf{b}_i$) by $\mathbf{a}_i[m]$ (respectively, $\mathbf{b}_i[m]$) for $1 \leq m \leq d-2$.

Then we can see that there exists such a product by solving a system of $2(d-2)$ homogeneous linear Diophantine equations of the form $B\mathbf{y} = \mathbf{0}$, where

$$B = \begin{pmatrix} \mathbf{a}_1[1] & \cdots & \mathbf{a}_r[1] \\ \vdots & \ddots & \vdots \\ \mathbf{a}_1[d-2] & \cdots & \mathbf{a}_r[d-2] \\ \mathbf{b}_1[1] & \cdots & \mathbf{b}_r[1] \\ \vdots & \ddots & \vdots \\ \mathbf{b}_1[d-2] & \cdots & \mathbf{b}_r[d-2] \end{pmatrix},$$

with an additional constraint that the numbers in the solution $\mathbf{y}$ that correspond to M_{i_1}

and M_{i_2} are non-zero since we must use these two matrices in the product. We repeat this process at most $r(r-1)$ times until we find a solution.

Hence, we can view the identity problem in $\mathrm{H}(d,\mathbb{Q})$ for $d \geq 3$ as the problem of solving systems of $2(d-2)$ homogeneous linear Diophantine equations with some constraints on the solution. As we can solve systems of linear Diophantine equations, we conclude that the identity problem in $\mathrm{H}(d,\mathbb{Q})$ is also decidable. □

7.4 The identity problem in matrix semigroups in dimension four

In this section, we prove that the identity problem is undecidable for 4×4 matrices, when the generating set has eight matrices, by introducing a new technique exploiting the anti-diagonal entries.

Theorem 7.20. *Given a semigroup S generated by eight 4×4 integer matrices, determining whether the identity matrix belongs to S is undecidable.*

Proof. We prove the claim by reducing from the PCP. We shall use an encoding to embed an instance of the PCP into a set of 4×4 integer matrices.

Let α be the mapping of Lemma 2.1 that maps elements of an arbitrary group alphabet into a binary group alphabet $\Gamma_2 = \{a, b, \bar{a}, \bar{b}\}$. We also define a monomorphism $f : \mathrm{FG}(\Gamma_2) \to \mathbb{Z}^{2\times2}$ as $f(a) = \left(\begin{smallmatrix}1 & 2\\ 0 & 1\end{smallmatrix}\right)$, $f(\bar{a}) = \left(\begin{smallmatrix}1 & -2\\ 0 & 1\end{smallmatrix}\right)$, $f(b) = \left(\begin{smallmatrix}1 & 0\\ 2 & 1\end{smallmatrix}\right)$ and $f(\bar{b}) = \left(\begin{smallmatrix}1 & 0\\ -2 & 1\end{smallmatrix}\right)$. Recall that the matrices $\left(\begin{smallmatrix}1 & 2\\ 0 & 1\end{smallmatrix}\right)$ and $\left(\begin{smallmatrix}1 & 0\\ 2 & 1\end{smallmatrix}\right)$ generate a free subgroup of $\mathrm{SL}(2,\mathbb{Z})$ [114]. The composition of two monomorphisms α and f gives us the embedding from an arbitrary group alphabet into the special linear group $\mathrm{SL}(2,\mathbb{Z})$. We use the composition of two monomorphisms α and f to encode a set of pairs of words over an arbitrary group alphabet into a set of 4×4 integer matrices in $\mathrm{SL}(4,\mathbb{Z})$ and denote it by β.

Let (g, h) be an instance of the PCP, where $g, h : \{a_1, \ldots, a_n\}^* \to \Sigma_2^*$, where $\Sigma_2 = \{a, b\}$. Without loss of generality, we can assume that the solution starts with the letter a_1. Moreover, we assume that this is the only occurence of a_1. We define the alphabet $\Gamma = \Sigma_2 \cup \Sigma_2^{-1} \cup \Sigma_B \cup \Sigma_B^{-1}$, where $\Sigma_B = \{q_0, q_1, p_0, p_1\}$ is the alphabet for the border letters that enforce the form of a solution.

Let us define the following sets of words $W_1 \cup W_2 \subseteq \mathrm{FG}(\Gamma) \times \mathrm{FG}(\Gamma)$, where

$$W_1 = \{(q_0 a \overline{q_0}, p_0 a \overline{p_0}),\ (q_0 b \overline{q_0}, p_0 b \overline{p_0}) \mid a, b \in \Sigma_2,\ \ q_0, p_0 \in \Sigma_B\} \text{ and}$$
$$W_2 = \left\{(q_0 \overline{g(a_1)} \overline{q_1}, p_0 \overline{h(a_1)} \overline{p_1}),\ (q_1 \overline{g(a_i)} \overline{q_1}, p_1 \overline{h(a_i)} \overline{p_1}) \mid 1 < i \leq n,\ \ q_0, q_1, p_0, p_1 \in \Sigma_B\right\}.$$

Intuitively, the words from set W_1 are used to construct words over Σ_2, and the words from set W_2 to cancel them according to the instance of the PCP.

Let us prove that $(q_0\overline{q_1}, p_0\overline{p_1}) \in \mathrm{FG}(W_1 \cup W_2)$ if and only if the PCP has a solution. It is easy to see that any pair of non-empty words in $\mathrm{FG}(W_1)$ is of the form $(q_0 w \overline{q_0}, p_0 w \overline{p_0})$ for $w \in \Sigma_2^+$. Then there exists a pair of words in $\mathrm{FG}(W_2)$ of the form $(q_0 \overline{w} \overline{q_1}, p_0 \overline{w} \overline{p_1})$ for some word $w \in \mathrm{FG}(\Gamma)$ if and only if the PCP has a solution. Therefore, the pair of words $(q_0\overline{q_1}, p_0\overline{p_1})$ can be constructed by concatenating pairs of words in W_1 and W_2 if and only if the PCP has a solution.

For each pair of words $(u, v) \in \mathrm{FG}(W_1 \cup W_2)$, we define a matrix $A_{u,v}$ to be $\left(\begin{smallmatrix} \beta(u) & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & \beta(v) \end{smallmatrix}\right) \in \mathrm{SL}(4, \mathbb{Z})$, where $\boldsymbol{O}_2$ is the zero matrix in $\mathbb{Z}^{2\times 2}$. Moreover, we define the following matrix

$$B_{q_1\overline{q_0}, p_1\overline{p_0}} = \begin{pmatrix} \boldsymbol{O}_2 & \beta(q_1\overline{q_0}) \\ \beta(p_1\overline{p_0}) & \boldsymbol{O}_2 \end{pmatrix} \in \mathrm{SL}(4, \mathbb{Z}).$$

Let S be a matrix semigroup generated by the set $\{A_{u,v}, B_{q_1\overline{q_0}, p_1\overline{p_0}} \mid (u, v) \in W_1 \cup W_2\}$. We already know that the pair $(q_0\overline{q_1}, p_0\overline{p_1})$ of words can be generated by concatenating words in W_1 and W_2 if and only if the PCP has a solution. The matrix semigroup S has the corresponding matrix $A_{q_0\overline{q_1}, p_0\overline{p_1}}$ and thus,

$$\begin{pmatrix} \beta(q_0\overline{q_1}) & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & \beta(p_0\overline{p_1}) \end{pmatrix} \begin{pmatrix} \boldsymbol{O}_2 & \beta(q_1\overline{q_0}) \\ \beta(p_1\overline{p_0}) & \boldsymbol{O}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{O}_2 & \beta(\varepsilon) \\ \beta(\varepsilon) & \boldsymbol{O}_2 \end{pmatrix} \in S.$$

Then we see that the identity matrix $\boldsymbol{I}_4$ exists in the semigroup S as follows:

$$\begin{pmatrix} \boldsymbol{O}_2 & \beta(\varepsilon) \\ \beta(\varepsilon) & \boldsymbol{O}_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{O}_2 & \beta(\varepsilon) \\ \beta(\varepsilon) & \boldsymbol{O}_2 \end{pmatrix} = \begin{pmatrix} \beta(\varepsilon) & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & \beta(\varepsilon) \end{pmatrix} = \begin{pmatrix} \boldsymbol{I}_2 & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & \boldsymbol{I}_2 \end{pmatrix} = \boldsymbol{I}_4 \in S.$$

Now we prove that the identity matrix does not exist in S if the PCP has no solution. It is easy to see that we cannot obtain the identity matrix only by multiplying 'A' matrices since there is no possibility of cancelling every border letter. We need to multiply the matrix $B_{q_1\overline{q_0}, p_1\overline{p_0}}$ with a product of 'A' matrices at some point to reach the identity matrix.

Note that the matrix $B_{q_1\overline{q_0},p_1\overline{p_0}}$ cannot be the first matrix of the product, followed by the 'A' matrices, because the upper right block of $B_{q_1\overline{q_0},p_1\overline{p_0}}$, which corresponds to the first word of the pair, should be multiplied with the lower right block of the 'A' matrix, which corresponds to the second word of the pair.

Suppose that the 'A' matrix is of form $\left(\begin{smallmatrix} \beta(q_0u\overline{q_1}) & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & \beta(p_0v\overline{p_1}) \end{smallmatrix}\right)$. Since the PCP instance has no solution, either u or v is not the empty word. We multiply $B_{q_1\overline{q_0},p_1\overline{p_0}}$ to the matrix and then obtain the following matrix:

$$\begin{pmatrix} \beta(q_0u\overline{q_1}) & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & \beta(p_0v\overline{p_1}) \end{pmatrix} \begin{pmatrix} \boldsymbol{O}_2 & \beta(q_1\overline{q_0}) \\ \beta(p_1\overline{p_0}) & \boldsymbol{O}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{O}_2 & \beta(q_0u\overline{q_0}) \\ \beta(p_0v\overline{p_0}) & \boldsymbol{O}_2 \end{pmatrix}.$$

We can see that either the upper right part or the lower left part cannot be $\beta(\varepsilon)$, which actually corresponds to the identity matrix in $\mathbb{Z}^{2\times 2}$. Now the only possibility of reaching the identity matrix is to multiply matrices which have $\mathrm{SL}(2,\mathbb{Z})$ matrices in the anti-diagonal coordinates like $B_{q_1\overline{q_0},p_1\overline{p_0}}$. However, we cannot cancel the parts because the upper right block (the lower left block) of the left matrix is multiplied with the lower left block (the upper right block) of the right matrix as follows:

$$\begin{pmatrix} \boldsymbol{O}_2 & A \\ B & \boldsymbol{O}_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{O}_2 & C \\ D & \boldsymbol{O}_2 \end{pmatrix} = \begin{pmatrix} AD & \boldsymbol{O}_2 \\ \boldsymbol{O}_2 & BC \end{pmatrix},$$

where A, B, C and D are matrices in $\mathbb{Z}^{2\times 2}$. As the first word of the pair is encoded in the upper right block of the matrix and the second word is encoded in the lower left block, it is not difficult to see that we cannot cancel the remaining blocks.

Currently, the undecidability bound for the PCP is five [122], and thus the semigroup S is generated by eight matrices. Recall that, in the beginning of the proof, we assumed that the letter a_1 of the PCP is used exacly once and is the first letter of a solution. This property is in fact present in [122]. □

Consider the membership problem called the *special diagonal membership problem*, where the task is to determine whether a scalar multiple of the identity matrix exists in a given matrix semigroup. The most recent undecidability bound is shown to be 14 by Halava et al. [81]. We improve the bound to eight, as the identity matrix is the only diagonal matrix of the semigroup S in the proof of Theorem 7.20. We also prove that the identity problem is undecidable in $\mathbb{H}(\mathbb{Q})^{2\times 2}$ as well by replacing the composition $f \circ \alpha$ of mappings with a mapping from a group alphabet to the set of rational quaternions [13].

Corollary 7.21. *Given a semigroup S generated by eight 4×4 integer matrices, determining whether there exists any diagonal matrix in S is undecidable.*

Corollary 7.22. *Given a semigroup S generated by eight 2×2 rational quaternion matrices, determining whether there exists the identity matrix S is undecidable.*

7.5 Concluding remarks and open problems

In this chapter, we considered reachability problems for polynomial iteration and matrix semigroups.

For polynomial iteration, we showed that, for one-dimensional polynomials, the problem is PSPACE-complete and for three-dimensional polynomials it is undecidable. The remaining case of two-dimensional polynomials remains open.

It would be interesting to see how the techniques of the proof can be applied to polynomials over rational numbers. Corollary 7.7 provides a lower bound for polynomials in the interval $[0, 1]$ but the upper bound is not clear as there are infinititely many rational numbers in the interval. It is possible that the p-adic norms used in similar settings in [24] can be useful to provide an upper bound or, at the very least, to prove decidability.

Recently, several variants of polynomial iteration have been considered by Ko [94]. He simplified the proof of the undecidability of three-dimensional polynomial iteration by reducing the PCP to iterating three-dimensional affine polynomials, that is, polynomials of the form $ax + b$. He also showed that if the set of polynomials $\mathcal{P}$ does not have polynomials of form $\pm x + b$, then the problem is PSPACE-complete in any dimension.

We also considered the identity problem in matrix semigroups and provided a better bound on the number of matrices in the generator set for 4×4 integer matrices, where the problem is undecidable. More importantly, we showed that there is no embedding of pairs of words into $\mathrm{SL}(3, \mathbb{Z})$. While this does not imply that the identity problem is decidable, it does provide hope as most of the undecidability proofs for matrix semigroups reduce from the PCP. We showed that the identity problem is decidable for Heisenberg groups. The natural follow-up question is whether other standard matrix problems, such as membership, are decidable in $\mathrm{H}(3, \mathbb{Z})$ or whether the identity problem is decidable for $\mathrm{H}(3, \mathbb{C})$.

Chapter 8

Summary

In this thesis, we studied several two-player turn-based games with respect to deciding which player has a winning strategy. Mostly, we considered games played on the integer lattice $\mathbb{Z}^d$, but also other, more exotic, arenas were considered. In several multidimensional games, we showed that the problem is undecidable. On the other hand, the decidability results fill-out the landscape for the considered games. The results are summarised in Table 8.1.

Perhaps the main result of the thesis is Corollary 5.13 as the model is very simple and could be easily used in more complex games. In fact, it has already been used to prove undecidability of existence of modular winning strategies in games with six-dimensional mean-payoff objectives [44] and a winning strategy in three-dimensional average-energy games [25].

The results of Chapter 3 provide an injective mapping that has a potential to be applied to other scenarios as well. The injective mappings are into different free groups, which limits its usability. Still, for example, a word game on pairs of words is very interesting as both initial and target configurations are the identity element.

In addition, we considered some related one-player systems. It would be interesting to consider their two-player variants. Actually, the identity problem for matrices is very closely related to the matrix games. Since the construction of the moves ensures that the target vector can be reached if and only if the identity matrix can be constructed by the turn-based multiplication. Similarly, polynomial iteration is closely connected to $\mathbb{Z}$-VAS games. Indeed, in $\mathbb{Z}$-VAS, the updates can be seen as polynomials of the form $p(x) = x + a$, which were called *counter polynomials* in [68]. Two-player games with polynomial updates

Game	Initial	Target	Complexity
Word game on FG(Γ)	u	ε	EXPTIME (Theorem 3.13)
Word game on FG(Γ) $\times \mathbb{Z}$	$[u, 0]$	$[\varepsilon, 0]$	undecidable (Theorem 3.9)
Word game on FG(Γ) $\times$ FG(Γ)	$[\varepsilon, \varepsilon]$	$[\varepsilon, \varepsilon]$	undecidable (Theorem 3.11)
Matrix game on $\mathbb{Z}^{4\times4}$	$\mathbf{x}_0$	$\mathbf{0}$	undecidable (Theorem 3.16)
Matrix game on $\mathbb{Z}^{3\times3}$	$\mathbf{x}_0$	$\mathbf{0}$	undecidable (Theorem 3.19)
Braid game on B_3	b_0	1	undecidable (Theorem 3.22)
Braid game on B_5	1	1	undecidable (Theorem 3.22)
$\mathbb{Z}$-VASS (Adam stateless, Eve any)	$[s_0, \dot{t}, z]$	$[s, \dot{t}, 0]$	EXPSPACE-complete (Theorem 4.4)
$\mathbb{Z}$-VASS (Adam any, Eve flat)	$[s_0, \dot{t_0}, z]$	$[s, \dot{t}, 0]$	EXPSPACE-complete (Corollary 4.17)
$\mathbb{Z}$-VASS (Adam flat, Eve stateless)	$[s, \dot{t_0}, z]$	$[s, \dot{t}, 0]$	EXPTIME-complete (Theorem 4.16)
$\mathbb{Z}^2$-VASS	$[s_0, \dot{t_0}, \mathbf{z}_0]$	$[s, \dot{t}, \mathbf{0}]$	undecidable (Corollary 5.6)[13]
$\mathbb{Z}^2$-VAS	$\mathbf{z}_0$	$\mathbf{0}$	undecidable (Corollary 5.13)
VAS (1-dim)	z	0	EXPTIME-complete (Theorem 4.19)
VAS (2-dim)	$\mathbf{z}_0$	$\mathbf{0}$	undecidable (Theorem 5.14)

Table 8.1: Summary of results of the thesis.

would be interesting in the case where there are no counter polynomials in players' move sets.

One of the most interesting directions is in Chapter 6. In which other scenarios can k-control be considered? We showed that, for a fixed k, the k-control $\mathbb{Z}^d$-VASS games are in NP, as opposed to EXPTIME when k is part of the input. Does limiting k by a polynomial with respect to the input yield interesting bounds?

While we considered the complexity of deciding which player has a winning strategy, we did not study the memory requirements for the strategies. For example, in the game

[13] Originally proven in [137].

constructed in Section 4.1 finite memory is sufficient for both players as by Lemmas 4.5, 4.6, 4.7 and 4.8, players' strategy depends on the degree of the vertex and the value of the counter modulo four. Are there arenas where strategies with finite memory are not sufficient?

We list some open problems related to the models studied in the thesis.

Open Problem 8.1. *Is the universality problem decidable for two-state weighted automata on infinite words?*

Open Problem 8.2. *Given a matrix game on vectors* $(A, E, \mathbf{x}_0, \mathbf{y})$*, where* $A, E \subseteq \mathbb{Z}^{2\times 2}$ *are the moves of Adam and Eve,* $\mathbf{x}_0, \mathbf{y} \in \mathbb{Z}^2$ *are the initial and target vectors. What is the complexity of deciding whether Eve has a winning strategy?*

Open Problem 8.3. *Given a block diagonal matrix game on vectors* $(A, E, \mathbf{x}_0, \mathbf{y})$*, where* $A, E \subseteq \mathbb{Z}^{3\times 3}$ *are the moves of Adam and Eve,* $\mathbf{x}_0, \mathbf{y} \in \mathbb{Z}^3$ *are the initial and target vectors, and each matrix in* A *and* E *is block diagonal. What is the complexity of deciding whether Eve has a winning strategy?*

Open Problem 8.4. *Given a braid game on* B_3*,* $(A, E, 1)$*, where* $A, E \subseteq B_3$ *are the moves of Adam and Eve, and the initial braid is the empty braid* 1*. What is the complexity of deciding whether Eve has a winning strategy?*

Open Problem 8.5. *Given a* $\mathbb{Z}$-VASS *game, where Eve is stateless and Adam has arbitrary state structure. What is the complexity of deciding whether Eve has a winning strategy?*

Open Problem 8.6. *Given an Attacker-Defender game with move sets of players* E *and* A*, for which it is undecidable to check for existence of a winning strategy. What is the smallest size of* A *and* E *such that the problem remains undecidable?*

While, the previous problem applies to most of the undecidable games present in the thesis, it is the most relevant for word games and $\mathbb{Z}^2$-VAS games. Also, the same question can be asked for games where deciding the winner is decidable as number of moves effects the complexity.

Open Problem 8.7. *Given a set of two-dimensional polynomials* $\mathcal{P} \subseteq \mathbb{Z}[x^2]$ *and* $\mathbf{x}_0$*. Is the two-dimensional polynomial iteration decidable?*

Open Problem 8.8. *Given* $S = \langle M_1, \ldots, M_r \rangle \subseteq \mathrm{H}(3, \mathbb{Q})$ *and* $M \in \mathrm{H}(3, \mathbb{Q})$*. Does* $M \in S$ *hold?*

Open Problem 8.9. *Given* $S = \langle M_1, \ldots, M_r \rangle \subseteq \mathrm{H}(3, \mathbb{C})$. *Does* $\boldsymbol{I}_3 \in S$ *hold?*

Last but not least, the identity problem for three-dimensional matrices:

Open Problem 8.10. *Given* $S = \langle M_1, \ldots, M_r \rangle \subseteq \mathbb{Z}^{3\times 3}$. *Does* $\boldsymbol{I}_3 \in S$ *hold?*

Bibliography

[1] Parosh Aziz Abdulla, Ahmed Bouajjani, and Julien d'Orso. Deciding monotonic games. In *Proceedings of CSL 2003*, volume 2803 of *LNCS*, pages 1–14. Springer, 2003. doi: 10.1007/978-3-540-45220-1_1.

[2] Parosh Aziz Abdulla, Ahmed Bouajjani, and Julien d'Orso. Monotonic and downward closed games. *Journal of Logic and Computation*, 18(1):153–169, 2008. doi: 10.1093/logcom/exm062.

[3] Parosh Aziz Abdulla, Richard Mayr, Arnaud Sangnier, and Jeremy Sproston. Solving parity games on integer vectors. In *Proceedings of CONCUR 2013*, volume 8052 of *LNCS*, pages 106–120. Springer, 2013. doi: 10.1007/978-3-642-40184-8_9.

[4] Andrei M. Akimenkov. Subgroups of the braid group B_4. *Mathematical notes of the Academy of Sciences of the USSR*, 50(6):1211–1218, 1991. doi: 10.1007/BF01158260.

[5] Natasha Alechina, Nils Bulling, Stephane Demri, and Brian Logan. On the complexity of resource-bounded logics. In *Proceedings of RP*, volume 9899 of *LNCS*, pages 36–50. Springer, 2016. doi: 10.1007/978-3-319-45994-3_3.

[6] Natasha Alechina, Nils Bulling, Brian Logan, and Hoang Nga Nguyen. The virtues of idleness: A decidable fragment of resource agent logic. *Artificial Intelligence*, 245:56–85, 2017. doi: 10.1016/j.artint.2016.12.005.

[7] Shaull Almagor, Udi Boker, and Orna Kupferman. What's decidable about weighted automata? In *Proceedings of ATVA 2011*, volume 6996 of *LNCS*, pages 482–491. Springer, 2011. doi: 10.1007/978-3-642-24372-1_37.

[8] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002. doi: 10.1145/585265.585270.

[9] Arjun Arul and Julien Reichert. The complexity of robot games on the integer line. In *Proceedings of QAPL 2013*, volume 117 of *EPTCS*, pages 132–148, 2013. doi: 10.4204/EPTCS.117.9.

[10] László Babai, Robert Beals, Jin-yi Cai, Gábor Ivanyos, and Eugene M. Luks. Multiplicative equations over commuting matrices. In *Proceedings of SODA 1996*, pages 498–507. SIAM, 1996. http://dl.acm.org/citation.cfm?id=313852.314109.

[11] Paul C. Bell, Mika Hirvensalo, and Igor Potapov. The identity problem for matrix semigroups in $\mathrm{SL}(2,\mathbb{Z})$ is NP-complete. In *Proceedings of SODA 2017*, pages 187–206. SIAM, 2017. doi: 10.1137/1.9781611974782.13.

[12] Paul C. Bell and Igor Potapov. On undecidability bounds for matrix decision problems. *Theoretical Computer Science*, 391(1-2):3–13, 2008. doi: 10.1016/j.tcs.2007.10.025.

[13] Paul C. Bell and Igor Potapov. Reachability problems in quaternion matrix and rotation semigroups. *Information and Computation*, 206(11):1353–1361, 2008. doi: 10.1016/j.ic.2008.06.004.

[14] Paul C. Bell and Igor Potapov. On the undecidability of the identity correspondence problem and its applications for word and matrix semigroups. *International Journal of Foundations of Computer Science*, 21(6):963–978, 2010. doi: 10.1142/S0129054110007660.

[15] Paul C. Bell and Igor Potapov. On the computational complexity of matrix semigroup problems. *Fundamenta Informaticae*, 116(1-4):1–13, 2012. doi: 10.3233/FI-2012-663.

[16] Amir M. Ben-Amram. Mortality of iterated piecewise affine functions over the integers: Decidability and complexity. *Computability*, 4(1):19–56, 2015. doi: 10.3233/COM-150032.

[17] Dietmar Berwanger and Erich Grädel. Fixed-point logics and solitaire games. *Theory of Computing Systems*, 37(6):675–694, 2004. doi: 10.1007/s00224-004-1147-5.

[18] Vladimir N. Bezverkhnii and Irina V. Dobrynina. Undecidability of the conjugacy problem for subgroups in the colored braid group R_5. *Matematicheskie Zametki*, 65(1):15–22, 1999. doi: 10.1007/BF02675004.

[19] Jean-Camille Birget and Stuart W. Margolis. Two-letter group codes that preserve aperiodicity of inverse finite automata. *Semigroup Forum*, 76:159–168, 2008. doi: 10.1007/s00233-007-9024-6.

[20] Kenneth R. Blaney and Andrey Nikolaev. A PTIME solution to the restricted conjugacy problem in generalized Heisenberg groups. *Groups Complexity Cryptology*, 8(1):69–74, 2016. doi: doi:10.1515/gcc-2016-0003.

[21] Vincent D. Blondel and Alexandre Megretski, editors. *Unsolved problems in mathematical systems and control theory*. Princeton University Press, 2004.

[22] Vincent D. Blondel and John N. Tsitsiklis. A survey of computational complexity results in systems and control. *Automatica*, 36(9):1249–1274, 2000. doi: 10.1016/S0005-1098(00)00050-9.

[23] Ahmed Bouajjani, Javier Esparza, and Oded Maler. Reachability analysis of pushdown automata: Application to model-checking. In *Proceedings of CONCUR 1997*, volume 1243, pages 135–150. Springer, 1997. doi: 10.1007/3-540-63141-0_10.

[24] Olivier Bournez, Oleksiy Kurganskyy, and Igor Potapov. Reachability problems for one-dimensional piecewise affine maps. Manuscript, 2017.

[25] Patricia Bouyer, Piotr Hofman, Nicolas Markey, Mickael Randour, and Martin Zimmermann. Bounding average-energy games. In *Proceedings of FoSSaCS 2017*, volume 10203 of *LNCS*, pages 179–195. Springer, 2017. doi: 10.1007/978-3-662-54458-7_11.

[26] Andrey Bovykin and Lorenzo Carlucci. Long games on braids. Manuscript, 2006.

[27] Tomás Brázdil, Václav Brozek, and Kousha Etessami. One-counter stochastic games. In *Proceedings of FSTTCS 2010*, volume 8 of *LIPIcs*, pages 108–119. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2010. doi: 10.4230/LIPIcs.FSTTCS.2010.108.

[28] Tomás Brázdil, Vojtech Forejt, Antonín Kucera, and Petr Novotný. Stability in graphs and games. In *Proceedings of CONCUR 2016*, volume 59 of *LIPIcs*, pages 10:1–10:14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. doi: 10.4230/LIPIcs.CONCUR.2016.10.

[29] Tomáš Brázdil, Petr Jančar, and Antonín Kučera. Reachability games on extended vector addition systems with states. In *Proceedings of ICALP 2010*, volume 6199 of *LNCS*, pages 478–489. Springer, 2010. doi: 10.1007/978-3-642-14162-1_40.

[30] Thomas Brihaye, Gilles Geeraerts, Axel Haddad, and Benjamin Monmege. To reach or not to reach? Efficient algorithms for total-payoff games. In *Proceedings of CONCUR 2015*, volume 42 of *LIPIcs*, pages 297–310. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. doi: 10.4230/LIPIcs.CONCUR.2015.297.

[31] Thomas Brihaye, Gilles Geeraerts, Axel Haddad, and Benjamin Monmege. Pseudopolynomial iterative algorithm to solve total-payoff games and min-cost reachability games. *Acta Informatica*, 54(1):85–125, 2017. doi: 10.1007/s00236-016-0276-z.

[32] Jean-Luc Brylinski. *Loop spaces, characteristic classes, and geometric quantization.* Birkhäuser, 1993.

[33] Cristian S. Calude, Sanjay Jain, Bakhadyr Khoussainov, Wei Li, and Frank Stephan. Deciding parity games in quasipolynomial time. In *Proceedings of STOC 2017*, pages 252–263. ACM, 2017. doi: 10.1145/3055399.3055409.

[34] Lorenzo Carlucci, Patrick Dehornoy, and Andreas Weiermann. Unprovability results involving braids. *Proceedings of the London Mathematical Society*, 102(1):159–192, 2011. doi: 10.1112/plms/pdq016.

[35] Julien Cassaigne, Vesa Halava, Tero Harju, and François Nicolas. Tighter undecidability bounds for matrix mortality, zero-in-the-corner problems, and more. *CoRR*, abs/1404.0644, 2014. https://arxiv.org/abs/1404.0644.

[36] Julien Cassaigne, Tero Harju, and Juhani Karhumäki. On the undecidability of freeness of matrix semigroups. *International Journal of Algebra and Computation*, 9(03n04):295–305, 1999. doi: 10.1142/S0218196799000199.

[37] Jakub Chaloupka. Z-reachability problem for games on 2-dimensional vector addition systems with states is in P. *Fundamenta Informaticae*, 123(1):15–42, 2013. doi: 10.3233/FI-2013-798.

[38] Krishnendu Chatterjee and Laurent Doyen. Energy parity games. *Theoretical Computer Science*, 458:49–60, 2012. doi: 10.1016/j.tcs.2012.07.038.

[39] Krishnendu Chatterjee, Laurent Doyen, Thomas A. Henzinger, and Jean-François Raskin. Generalized mean-payoff and energy games. In *Proceedings of FSTTCS 2010*, volume 8 of *LIPIcs*, pages 505–516. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2010. doi: 10.4230/LIPIcs.FSTTCS.2010.505.

[40] Krishnendu Chatterjee, Laurent Doyen, Mickael Randour, and Jean-François Raskin. Looking at mean-payoff and total-payoff through windows. *Information and Computation*, 242:25–52, 2015. doi: 10.1016/j.ic.2015.03.010.

[41] Krishnendu Chatterjee, Mickael Randour, and Jean-François Raskin. Strategy synthesis for multi-dimensional quantitative objectives. In *Proceedings of CONCUR 2012*, volume 7454 of *LNCS*, pages 115–131. Springer, 2012. doi: 10.1007/978-3-642-32940-1_10.

[42] Krishnendu Chatterjee, Mickael Randour, and Jean-François Raskin. Strategy synthesis for multi-dimensional quantitative objectives. *Acta Informatica*, 51(3):129–163, 2014. doi: 10.1007/s00236-013-0182-6.

[43] Krishnendu Chatterjee and Yaron Velner. Hyperplane separation technique for multidimensional mean-payoff games. In *Proceedings of CONCUR 2013*, volume 8052 of *LNCS*, pages 500–515. Springer, 2013. doi: 10.1007/978-3-642-40184-8_35.

[44] Krishnendu Chatterjee and Yaron Velner. Hyperplane separation technique for multidimensional mean-payoff games. *Journal of Computer and System Sciences*, 88:236–259, 2017. doi: 10.1016/j.jcss.2017.04.005.

[45] Christian Choffrut and Juhani Karhumäki. Some decision problems on integer matrices. *RAIRO - Theoretical Informatics and Applications*, 39(1):125–131, 2005. doi: 10.1051/ita:2005007.

[46] Ventsislav Chonev, Joël Ouaknine, and James Worrell. The orbit problem in higher dimensions. In *Proceedings of STOC 2013*, pages 941–950. ACM, 2013. doi: 10.1145/2488608.2488728.

[47] Ventsislav Chonev, Joël Ouaknine, and James Worrell. The polyhedron-hitting problem. In *Proceedings of SODA 2015*, pages 940–956. SIAM, 2015. doi: 10.1137/1.9781611973730.64.

[48] Ventsislav Chonev, Joël Ouaknine, and James Worrell. On the complexity of the orbit problem. *Journal of the ACM*, 63(3):23:1–23:18, 2016. doi: 10.1145/2857050.

[49] Volker Claus. Some remarks on PCP(k) and related problems. *Bulletin of EATCS*, 12:54–61, 1980.

[50] Thomas Colcombet, Marcin Jurdziński, Ranko Lazić, and Sylvain Schmitz. Perfect half space games. In *Proceedings of LICS 2017*, pages 1–11. IEEE, 2017. doi: 10.1109/LICS.2017.8005105.

[51] Graham P. Collins. Computing with quantum knots. *Scientific American*, 294(4):56–63, 2006. doi: 10.1038/scientificamerican0406-56.

[52] Hubert Comon and Véronique Cortier. Flatness is not a weakness. In *Proceedings of CSL 2000*, volume 1862 of *LNCS*, pages 262–276. Springer, 2000. doi: 10.1007/3-540-44622-2_17.

[53] Hubert Comon and Yan Jurski. Multiple counters automata, safety analysis and Presburger arithmetic. In *Proceedings of CAV, 1998*, volume 1427 of *LNCS*, pages 268–279. Springer, 1998. doi: 10.1007/BFb0028751.

[54] Hubert Comon and Yan Jurski. Timed automata and the theory of real numbers. In *Proceedings of CONCUR 1999*, volume 1664 of *LNCS*, pages 242–257. Springer, 1999. doi: 10.1007/3-540-48320-9_18.

[55] Marston Conder, Edmund Robertson, and Peter Williams. Presentations for 3-dimensional special linear groups over integer rings. *Proceedings of the American Mathematical Society*, 115(1):19–26, 1992. doi: 10.2307/2159559.

[56] Marston D. E. Conder. Some unexpected consequences of symmetry computations. In *SIGMAP 2014*, volume 159 of *PROMS*, pages 71–79. Springer, 2016. doi: 10.1007/978-3-319-30451-9_3.

[57] Jean-Baptiste Courtois and Sylvain Schmitz. Alternating vector addition systems with states. In *Proceedings of MFCS 2014*, volume 8634 of *LNCS*, pages 220–231. Springer, 2014. doi: 10.1007/978-3-662-44522-8_19.

[58] Felipe Cucker, Pascal Koiran, and Steve Smale. A polynomial time algorithm for Diophantine equations in one variable. *Journal of Symbolic Computation*, 27(1):21–29, 1999. doi: 10.1006/jsco.1998.0242.

[59] Patrick Dehornoy, Ivan Dynnikov, Dale Rolfsen, and Bert Wiest. *Ordering braids*, volume 148 of *Mathematical Surveys and Monographs*. Mathematical Surveys and Monographs, 2008.

[60] Christoph Dittmann, Stephan Kreutzer, and Alexandru I. Tomescu. Graph operations on parity games and polynomial-time algorithms. *Theoretical Computer Science*, 614:97–108, 2016. doi: 10.1016/j.tcs.2015.11.044.

[61] Jing Dong and Qinghui Liu. Undecidability of infinite Post correspondence problem for instances of size 8. *RAIRO - Theoretical Informatics and Applications*, 46(3):451–457, 2012. doi: 10.1051/ita/2012015.

[62] Laurent Doyen and Alexander Rabinovich. Robot games. Technical Report LSV-13-02, LSV, ENS Cachan, 2013. http://www.lsv.fr/Publis/RAPPORTS_LSV/PDF/rr-lsv-2013-02.pdf.

[63] Catherine Dufourd, Alain Finkel, and Philippe Schnoebelen. Reset nets between decidability and undecidability. In *Proceedings of ICALP 1998*, volume 1443 of *LNCS*, pages 103–115. Springer, 1998. doi: 10.1007/BFb0055044.

[64] E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *Proceedings of FOCS 1991*, pages 368–377. IEEE, 1991. doi: 10.1109/SFCS.1991.185392.

[65] E. Allen Emerson, Charanjit S. Jutla, and A. Prasad Sistla. On model checking for the μ-calculus and its fragments. *Theoretical Computer Science*, 258(1-2):491–522, 2001. doi: 10.1016/S0304-3975(00)00034-7.

[66] David B. A. Epstein, Michael S. Paterson, James W. Cannon, Darek F. Holt, Silvio V. Levy, and William P. Thurston. *Word processing in groups*. AK Peters, Ltd., 1992.

[67] Uli Fahrenberg, Line Juhl, Kim G. Larsen, and Jirí Srba. Energy games in multi-weighted automata. In *Proceedings of ICTAC 2011*, volume 6916 of *LNCS*, pages 95–115. Springer, 2011. doi: 10.1007/978-3-642-23283-1_9.

[68] Alain Finkel, Stefan Göller, and Christoph Haase. Reachability in register machines with polynomial updates. In *Proceedings of MFCS 2013*, volume 8087 of *LNCS*, pages 409–420. Springer, 2013. doi: 10.1007/978-3-642-40313-2_37.

[69] Michael J. Fischer and Michael O. Rabin. Super-exponential complexity of Presburger arithmetic. In *Complexity of Computation, SIAM-AMS Proceedings*, volume 7, pages 27–41. SIAM, 1974. doi: 10.1007/978-3-7091-9459-1_5.

[70] Esther Galby, Joël Ouaknine, and James Worrell. On matrix powering in low dimensions. In *Proceedings of STACS 2015*, volume 30 of *LIPIcs*, pages 329–340, 2015. doi: 10.4230/LIPIcs.STACS.2015.329.

[71] Aniruddh Gandhi, Bakhadyr Khoussainov, and Jiamou Liu. Efficient algorithms for games played on trees with back-edges. *Fundamenta Informaticae*, 111(4):391–412, 2011. doi: 10.3233/FI-2011-569.

[72] David Garber. Braid group cryptography. In *Braids: Introductory lectures on braids, configurations and their applications*, volume 19, pages 329–403. World Scientific, 2010. doi: 10.1142/9789814291415_0006.

[73] Stephane Gaubert and Ricardo Katz. Reachability problems for products of matrices in semirings. *IJAC*, 16(3):603–627, 2006. doi: 10.1142/S021819670600313X.

[74] Razvan Gelca and Alejandro Uribe. From classical theta functions to topological quantum field theory. In *The influence of Solomon Lefschetz in geometry and topology*, volume 621 of *Contemprorary Mathematics*, pages 35–68. American Mathematical Society, 2014. doi: 10.1090/conm/621.

[75] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, logics, and infinite games: A guide to current research*, volume 2500 of *LNCS*. Springer, 2002. doi: 10.1007/3-540-36387-4.

[76] Yuri Gurevich and Paul Schupp. Membership problem for the modular group. *SIAM Journal of Computing*, 37(2):425–459, 2007. doi: 10.1137/050643295.

[77] Christoph Haase and Simon Halfon. Integer vector addition systems with states. In *Proceedings of RP 2014*, volume 8762 of *LNCS*, pages 112–124. Springer, 2014. doi: 10.1007/978-3-319-11439-2_9.

[78] Vesa Halava. *Finite substitutions and integer weighted finite automata.* Licentiate thesis, University of Turku, Turku, Finland, 1998. http://tucs.fi/publications/view/?pub_id=licHalava98a.

[79] Vesa Halava and Tero Harju. Undecidability in integer weighted finite automata. *Fundamenta Informaticae*, 38(1-2):189–200, 1999. doi: 10.3233/FI-1999-381215.

[80] Vesa Halava and Tero Harju. Undecidability of infinite post correspondence problem for instances of size 9. *ITA*, 40(4):551–557, 2006. doi: 10.1051/ita:2006039.

[81] Vesa Halava, Tero Harju, and Mika Hirvensalo. Undecidability bounds for integer matrices using Claus instances. *International Journal of Foundations of Computer Science*, 18(5):931–948, 2007. doi: 10.1142/S0129054107005066.

[82] Vesa Halava, Tero Harju, Reino Niskanen, and Igor Potapov. Weighted automata on infinite words in the context of Attacker-Defender games. In *Proceedings of CiE 2015*, volume 9136 of *LNCS*, pages 206–215. Springer, 2015. doi: 10.1007/978-3-319-20028-6_21.

[83] Vesa Halava, Tero Harju, Reino Niskanen, and Igor Potapov. Weighted automata on infinite words in the context of Attacker-Defender games. *Information and Computation*, 255:27–44, 2017. doi: 10.1016/j.ic.2017.05.001.

[84] Vesa Halava and Mika Hirvensalo. Improved matrix pair undecidability results. *Acta Informatica*, 44(3):191–205, 2007. doi: 10.1007/s00236-007-0047-y.

[85] Vesa Halava, Yuri Matiyasevich, and Reino Niskanen. Small semi-Thue system universal with respect to the termination problem. *Fundamenta Informaticae*, 154(1-4):177–184, 2017. doi: 10.3233/FI-2017-1559.

[86] Vesa Halava, Reino Niskanen, and Igor Potapov. On robot games of degree two. In *Proceedings of LATA 2015*, volume 8977 of *LNCS*, pages 224–236. Springer, 2015. doi: 10.1007/978-3-319-15579-1_17.

[87] Juha Honkala. A Kraft-McMillan inequality for free semigroups of upper-triangular matrices. *Information and Computation*, 239:216–221, 2014. doi: 10.1016/j.ic.2014.09.002.

[88] Paul Hunter. Reachability in succinct one-counter games. In *Proceedings of RP 2015*, volume 9328 of *LNCS*, pages 37–49. Springer, 2015. doi: 10.1007/978-3-319-24537-9_5.

[89] Paul Hunter and Jean-François Raskin. Quantitative games with interval objectives. In *Proceedings of FSTTCS 2014*, volume 29 of *LIPIcs*, pages 365–377. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014. doi: 10.4230/LIPIcs.FSTTCS.2014.365.

[90] Marcin Jurdziński. Deciding the winner in parity games is in UP ∩ co-UP. *Information Processing Letters*, 68(3):119–124, 1998. doi: 10.1016/S0020-0190(98)00150-1.

[91] Marcin Jurdziński. Small progress measures for solving parity games. In *Proceedings of STACS 2000*, volume 1770 of *LNCS*, pages 290–301. Springer, 2000. doi: 10.1007/3-540-46541-3_24.

[92] Marcin Jurdziński, Ranko Lazić, and Sylvain Schmitz. Fixed-dimensional energy games are in pseudo-polynomial time. In *Proceedings of ICALP 2015*, volume 9135 of *LNCS*, pages 260–272. Springer, 2015. doi: 10.1007/978-3-662-47666-6_21.

[93] Ravi Kannan and Richard J. Lipton. Polynomial-time algorithm for the orbit problem. *Journal of the ACM*, 33(4):808–821, 1986. doi: 10.1145/6490.6496.

[94] Sang-Ki Ko. Personal communications, 2017.

[95] Sang-Ki Ko, Reino Niskanen, and Igor Potapov. On the identity problem for the special linear group and the Heisenberg group. *CoRR*, abs/1706.04166, 2017. https://arxiv.org/abs/1706.04166.

[96] Sang-Ki Ko and Igor Potapov. Composition problems for braids: Membership, identity and freeness. *CoRR*, abs/1707.08389, 2017. http://arxiv.org/abs/1707.08389.

[97] Pascal Koiran, Michel Cosnard, and Max H. Garzon. Computability with low-dimensional dynamical systems. *Theoretical Computer Science*, 132(2):113–128, 1994. doi: 10.1016/0304-3975(94)90229-1.

[98] Daniel König, Markus Lohrey, and Georg Zetzsche. Knapsack and subset sum problems in nilpotent, polycyclic, and co-context-free groups. *Algebra and Computer Science*, 677:138–153, 2016. doi: 10.1090/conm/677/13625.

[99] Ivan Korec. Small universal register machines. *Theoretical Computer Science*, 168(2):267–301, 1996. doi: 10.1016/S0304-3975(96)00080-1.

[100] Bertram Kostant. Quantization and unitary representations. In *Lectures in Modern Analysis and Applications III*, pages 87–208. Springer, 1970. doi: 10.1007/BFb0079068.

[101] Michal Kunc. Regular solutions of language inequalities and well quasi-orders. *Theoretical Computer Science*, 348(2-3):277–293, 2005. doi: 10.1016/j.tcs.2005.09.018.

[102] Michal Kunc. The power of commuting with finite sets of words. *Theory Computing Systems*, 40(4):521–551, 2007. doi: 10.1007/s00224-006-1321-z.

[103] Orna Kupferman, Moshe Y. Vardi, and Pierre Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, 2000. doi: 10.1145/333979.333987.

[104] Oleksiy Kurganskyy and Igor Potapov. Reachability problems for PAMs. In *Proceedings of SOFSEM 2016*, volume 9587 of *LNCS*, pages 356–368. Springer, 2016. doi: 10.1007/978-3-662-49192-8_29.

[105] Oleksiy Kurganskyy, Igor Potapov, and Fernando Sancho-Caparrini. Reachability problems in low-dimensional iterative maps. *International Journal of Foundations of Computer Science*, 19(4):935–951, 2008. doi: 10.1142/S0129054108006054.

[106] Sige-Yuki Kuroda. Classes of languages and linear-bounded automata. *Information and Control*, 7(2):207–223, 1964. doi: 10.1016/s0019-9958(64)90120-2.

[107] Peter S. Landweber. Three theorems on phrase structure grammars of type 1. *Information and Control*, 6(2):131–136, 1963. doi: 10.1016/s0019-9958(63)90169-4.

[108] Jérôme Leroux, Vincent Penelle, and Grégoire Sutre. The context-freeness problem is coNP-complete for flat counter systems. In *Proceedings of ATVA 2014*, volume 8837 of *LNCS*, pages 248–263. Springer, 2014. doi: 10.1007/978-3-319-11936-6_19.

[109] Jérôme Leroux and Sylvain Schmitz. Demystifying reachability in vector addition systems. In *Proceedings of LICS 2015*, pages 56–67. IEEE, 2015. doi: 10.1109/LICS.2015.16.

[110] Jérôme Leroux and Grégoire Sutre. On flatness for 2-dimensional vector addition systems with states. In *Proceedings of CONCUR 2004*, volume 3170 of *LNCS*, pages 402–416. Springer, 2004. doi: 10.1007/978-3-540-28644-8_26.

[111] Jérôme Leroux and Grégoire Sutre. Flat counter automata almost everywhere! In *Proceedings of ATVA 2005*, volume 3707 of *LNCS*, pages 489–503. Springer, 2005. doi: 10.1007/11562948_36.

[112] Alexei Lisitsa and Igor Potapov. Membership and reachability problems for row-monomial transformations. In *Proceedings of MFCS 2004*, volume 3153 of *LNCS*, pages 623–634. Springer, 2004. doi: 10.1007/978-3-540-28629-5_48.

[113] Olivier Ly and Zhilin Wu. On effective construction of the greatest solution of language inequality XA $\subseteq$ BX. *Theoretical Computer Science*, 528:12–31, 2014. doi: 10.1016/j.tcs.2014.02.001.

[114] Roger C. Lyndon and Paul E. Schupp. *Combinatorial group theory*. Springer, 1977. doi: 10.1007/978-3-642-61896-3.

[115] Andrei A. Markov. On certain insoluble problems concerning matrices. *Doklady Akademii Nauk SSSR*, 57(6):539–542, 1947.

[116] Donald A. Martin. Borel determinacy. *Annals of Mathematics*, 102(2):363–371, 1975. doi: 10.2307/1971035.

[117] Yuri Matiyasevich and Géraud Sénizergues. Decision problems for semi-Thue systems with a few rules. *Theoretical Computer Science*, 330(1):145–169, 2005. doi: 10.1016/j.tcs.2004.09.016.

[118] Ernst W. Mayr. An algorithm for the general Petri net reachability problem. In *Proceedings of STOC 1981*, pages 238–246. ACM, 1981. doi: 10.1145/800076.802477.

[119] Robert McNaughton. Infinite games played on finite graphs. *Annals of Pure and Applied Logic*, 65(2):149–184, 1993. doi: 10.1016/0168-0072(93)90036-D.

[120] Marvin L. Minsky. *Computation: finite and infinite machines*. Prentice-Hall, Inc., 1967. https://dl.acm.org/citation.cfm?id=1095587.

[121] Alexei Mishchenko and Alexander Treier. Knapsack problem for nilpotent groups. *Groups Complexity Cryptology*, 9(1):87–98, 2017. doi: 10.1515/gcc-2017-0006.

[122] Turlough Neary. Undecidability in binary tag systems and the Post correspondence problem for five pairs of words. In *Proceedings of STACS 2015*, volume 30 of *LIPIcs*, pages 649–661. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015. doi: 10.4230/LIPIcs.STACS.2015.649.

[123] Reino Niskanen. Robot games with states in dimension one. In *Proceedings of RP 2016*, volume 9899 of *LNCS*, pages 163–176. Springer, 2016. doi: 10.1007/978-3-319-45994-3_12.

[124] Reino Niskanen. Reachability problem for polynomial iteration is PSPACE-complete. In *Proceedings of RP 2017*, volume 10506 of *LNCS*, pages 132–143. Springer, 2017. doi: 10.1007/978-3-319-67089-8_10.

[125] Reino Niskanen, Igor Potapov, and Julien Reichert. Undecidability of two-dimensional robot games. In *Proceedings of MFCS 2016*, volume 58 of *LIPIcs*, pages 73:1–73:13. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. doi: 10.4230/LIPIcs.MFCS.2016.73.

[126] Joël Ouaknine, João Sousa Pinto, and James Worrell. On termination of integer linear loops. In *Proceedings of SODA 2015*, pages 957–969. SIAM, 2015. doi: 10.1137/1.9781611973730.65.

[127] Joël Ouaknine and James Worrell. On the positivity problem for simple linear recurrence sequences. In *Proceedings of ICALP 2014*, volume 8573 of *LNCS*, pages 318–329. Springer, 2014. doi: 10.1007/978-3-662-43951-7_27.

[128] Joël Ouaknine and James Worrell. Ultimate positivity is decidable for simple linear recurrence sequences. In *Proceedings of ICALP 2014*, volume 8573 of *LNCS*, pages 330–341. Springer, 2014. doi: 10.1007/978-3-662-43951-7_28.

[129] Prakash Panangaden and Éric Oliver Paquette. A categorical presentation of quantum computation with anyons. In *New structures for Physics*, volume 813 of *LNP*, pages 983–1025. Springer, 2011. doi: 10.1007/978-3-642-12821-9_15.

[130] Michael S. Paterson. Unsolvability in 3×3 matrices. *Studies in Applied Mathematics*, 49(1):105, 1970. doi: 10.1002/sapm1970491105.

[131] Emil L. Post. A variant of a recursively unsolvable problem. *Bulletin of the American Mathematical Society*, 52(4):264–268, 1946.

[132] Igor Potapov. From Post systems to the reachability problems for matrix semigroups and multicounter automata. In *Proceedings of DLT 2004*, volume 3340 of *LNCS*, pages 345–356. Springer, 2004. doi: 10.1007/978-3-540-30550-7_29.

[133] Igor Potapov. Composition problems for braids. In *Proceedings of FSTTCS 2013*, volume 24 of *LIPIcs*, pages 175–187. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2013. doi: 10.4230/LIPIcs.FSTTCS.2013.175.

[134] Igor Potapov and Pavel Semukhin. Decidability of the membership problem for 2×2 integer matrices. In *Proceedings of SODA 2017*, pages 170–186. SIAM, 2017. doi: 10.1137/1.9781611974782.12.

[135] Igor Potapov and Pavel Semukhin. Membership problem in $\mathrm{GL}(2,\mathbb{Z})$ extended by singular matrices. In *Proceedings of MFCS 2017*, volume 83 of *LIPIcs*, pages 44:1–44:13. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. doi: 10.4230/LIPIcs.MFCS.2017.44.

[136] Julien Reichert. On the complexity of counter reachability games. In *Proceedings of RP 2013*, volume 8169 of *LNCS*, pages 196–208. Springer, 2013. doi: 10.1007/978-3-642-41036-9_18.

[137] Julien Reichert. *Reachability games with counters: Decidability and algorithms*. Doctoral thesis, Laboratoire Spécification et Vérification, ENS Cachan, France, 2015.

[138] Julien Reichert. On the complexity of counter reachability games. *Fundamenta Informaticae*, 143(3-4):415–436, 2016. doi: 10.3233/FI-2016-1320.

[139] Emmanuel Roche and Yves Schabes. Speech recognition by composition of weighted finite automata. In *Finite-state language processing*, A Bradford book, pages 431–454. MIT press, 1997.

[140] Jean-François Romeuf. A polynomial algorithm for solving systems of two linear Diophantine equations. *Theoretical Computer Science*, 74(3):329–340, 1990. doi: 10.1016/0304-3975(90)90082-s.

[141] Keijo Ruohonen. Reversible machines and Post's correspondence problem for biprefix morphisms. *Elektronische Informationsverarbeitung und Kybernetik*, 21(12):579–595, 1985.

[142] Sven Schewe. From parity and payoff games to linear programming. In *Proceedings of MFCS 2009*, volume 5734 of *LNCS*, pages 675–686. Springer, 2009. doi: 10.1007/978-3-642-03816-7_57.

[143] Sven Schewe. Solving parity games in big steps. *Journal of Computer and System Sciences*, 84:243–262, 2017. doi: 10.1016/j.jcss.2016.10.002.

[144] Dejvuth Suwimonteerabuth, Stefan Schwoon, and Javier Esparza. Efficient algorithms for alternating pushdown systems with an application to the computation of certificate chains. In *Proceedings of ATVA 2006*, volume 4218 of *LNCS*, pages 141–153. Springer, 2006. doi: 10.1007/11901914_13.

[145] Yaron Velner. Robust multidimensional mean-payoff games are undecidable. In *Proceedings of FoSSaCS 2015*, volume 9034 of *LNCS*, pages 312–327. Springer, 2015. doi: 10.1007/978-3-662-46678-0_20.

[146] Igor Walukiewicz. Pushdown processes: Games and model-checking. *Information and Computation*, 164(2):234–263, 2001. doi: 10.1006/inco.2000.2894.

[147] Thomas Wilke. Alternating tree automata, parity games, and modal μ-calculus. *Bulletin of the Belgian Mathematical Society*, 8(2):359–391, 2001.

[148] Wieslaw Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200(1-2):135–183, 1998. doi: 10.1016/S0304-3975(98)00009-7.

[149] Uri Zwick and Michael S. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1&2):343–359, 1996. doi: 10.1016/0304-3975(95)00188-3.

Index