Reachability Games with Counters: Decidability and Algorithms

Thèse de doctorat
présentée par

Julien Reichert

à
l'École Normale Supérieure de Cachan

en vue de l'obtention du grade de
Docteur en Informatique

et soutenue à Cachan le 30 juillet 2015 devant le jury composé de :

Béatrice Bérard	Présidente du jury
Dietmar Berwanger	Directeur de thèse
Hubert Comon-Lundh	Examinateur
Catalin Dima	Examinateur
Laurent Doyen	Directeur de thèse
Jérôme Leroux	Rapporteur
Igor Potapov	Rapporteur

Laboratoire Spécification et Vérification (LSV) ; École Normale Supérieure de Cachan
61, avenue du Président Wilson ; 94235 CACHAN Cedex, France

Remerciements

À l'heure où je termine la rédaction de ce manuscrit, il me revient une scène datant d'automne 2005, au réfectoire du lycée Kléber. Alors que j'avais juste annoncé ma volonté de suivre l'option informatique, une connaissance en MP m'a annoncé que l'informatique ne consistait pas qu'à programmer des jeux vidéos. Loin s'en fallut effectivement, quoique la programmation de jeux prît une partie de mon temps libre durant les années qui suivirent. Ce même mot, « jeux », ne pouvait que m'intriguer, de même que le vocabulaire qui lui était associé, notamment les stratégies, et je me suis inscrit avec enthousiasme au cours du MPRI en question, afin d'étudier la théorie de ce que j'appliquais en pratique devant un plateau ou cartes en main.

C'est également en prépa qu'on trouve la source de ma vocation principale, à savoir l'enseignement. Pendant toute ma scolarité, j'ai eu la chance d'avoir régulièrement d'excellents instituteurs et professeurs, en fait bien trop pour que je les cite tous, mais je me dois de préciser que ce sont deux professeurs d'histoire, Bernard Robin et Christian Cropsal, qui m'ont montré que la transmission de connaissances pouvait être un loisir tant pour l'enseignant que pour l'élève. Mais revenons en prépa. J'ai passé une première année formidable, grâce aux encouragements de Frédéric Cuvellier et Thierry Meyer, et je remercie ce dernier pour la confiance qu'il m'a accordée, alors même que mon niveau en sciences physiques subissait un effondrement dont il ne s'est à ce jour pas encore remis (je parle de mon niveau), en me permettant de passer en MP*, où j'ai été contaminé par ce qui semble être connu sous le nom du syndrôme de « je veux faire l'ENS en info ». Cette intégration à l'ENS, sur une sombre histoire de minima locaux dans une matrice, m'a permis d'apprendre l'informatique d'une part, mais surtout de passer l'agrégation d'autre part, toujours dans des conditions idéales. Mes remerciements vont à l'équipe pédagogique du département informatique de l'école, notamment à mon tuteur Thomas Chatain concernant le cursus et en particulier la recherche de stages, ainsi qu'à Claudine Picaronny concernant la préparation à l'agrégation. J'ajoute une mention de dernière minute pour Serge Abiteboul, grâce à qui j'ai pu prendre contact avec de nombreux professeurs bienveillants en vue de me *prépa*rer pour mon futur poste d'enseignement que j'ai l'honneur et la joie de prendre à la rentrée.

Puisque je m'étais mis en tête dès mon arrivée à l'école que je voulais enseigner, j'ai certes fait peu de cas de l'ouverture au monde de la recherche qui m'était proposée. Cependant, avant le début de ma thèse, de nombreuses personnes m'ont permis de voir que ce monde-là était tout aussi enrichissant que celui dans lequel je suis entré en septembre 2014.
Il s'agit tout d'abord des équipes "Logik und Theorie diskreter Systeme" et "Mathematische Grundlagen der Informatik" de l'université d'Aix-la-Chapelle, où j'ai fait un stage court en juin et juillet 2008. Danke sehr an Erich Grädel, mit der Hilfe von Łukasz Kaiser für mein erstes Praktikum, das ich in ganz guten Bedingungen verbracht habe. Vielen Dank auch an Wolfgang Thomas und an beiden Teamen für alle interessante Gespräche.
L'année suivante, c'était plus qu'un premier contact qui m'attendait, au cours de mon stage de M1 à Bruxelles. J'ai été accueilli dans une équipe dynamique et encadré par Jean-François Raskin, ainsi que Gilles Geeraerts. Ce fut pour moi une occasion de plus de côtoyer des chercheurs sympathiques et intéressants, pour ne citer que Emmanuel, Naiyong, Nicolas, Raffaella, Thierry et Tristan, dont beaucoup que j'ai eu la chance de croiser à nouveau par la suite.
Il manque à ces deux paragraphes deux noms, ceux de mes directeurs de thèse, Dietmar Ber-

WANGER et Laurent DOYEN, rencontrés respectivement à Aix-la-Chapelle et Bruxelles et arrivés au LSV entre-temps. Je souhaite exprimer ici toute ma gratitude pour m'avoir permis de faire une thèse sur une double thématique, les domaines qui m'avaient le plus intéressé parmi mes cours de M2, dans un lieu agréable, avec et au côté de personnes que je connaissais et que j'appréciais. Je retiendrai notamment que j'ai appris en trois ans des choses que je ne m'attendais pas à apprendre ici, notamment en ce qui concerne la façon de présenter des résultats de recherche, qui s'apparente à la pédagogie et qui me faisait visiblement le plus défaut. Il a fallu de la patience pour m'aider à passer de la copie d'examen à un texte censé faire office de référence pour de futurs travaux, et me défaire des écueils courants que je ne manquais pas de reproduire, paragraphe après paragraphe, version après version, dans mes textes. En outre, être un bon chercheur, ce n'est pas que trouver les réponses, fussent-elles partielles, mais aussi trouver les questions, et ceci restera mon point faible au moment où cette branche de carrière s'arrête.

I would like to express special thanks to Jérôme LEROUX and Igor POTAPOV for their review work on my thesis, and also for the nice time I had at the RP conferences. Also many thanks to Béatrice BÉRARD, Hubert COMON and Catalin DIMA for their participation to the jury.

Je remercie encore mon entourage au LSV, avec une mention particulière aux locataires du quatrième étage, notamment ceux dont j'ai temporairement partagé le bureau : Alban, Sandie, Szymon, Hernán, Gonzalo, Loïg, Simon et Georgios. I also thank Arjun, who worked with me on my first article and whom I hope to see again, as well as Marie and Anup, who worked with my on my last proofs. Danke an alle deutsche Forscher, aber auch und überhaupt Skatfreunde : Christoph, Jana und Stefan. J'ai eu beaucoup de plaisir à faire mon monitorat à l'ENS, principalement pour les élèves de première année du département informatique, et j'espère que ce plaisir était partagé. Je souhaiterais nommer des élèves de la première heure, avec qui j'ai gardé de bons contacts : Yann et Clément, ce dernier m'ayant poussé dans mes derniers retranchements en ce qui concerne le besoin d'être totalement rigoureux, ce qui fait qu'il est devenu bien plus difficile de me prendre au dépourvu grâce à lui. Ces trois années ont aussi passé à une vitesse exceptionnelle grâce aux trois clubs qui ont pu compter sur mon assiduité : bridge, skat et opéra. Autant de sources de bons moments, complétés par des soirées passées à jouer ou à discuter de choses et d'autres, avec quatre personnes que je citerais entre tous : Éric, Marie-Clémentine, Michaël et Sylvain.

Je conclus ces remerciements en les adressant aux personnes qui me sont le plus chères : mes parents et mes grands-parents, qui n'ont jamais cessé de me soutenir tout au long de ma scolarité et qui continueront alors même que celle-ci s'achève.

Abstract

In this thesis, we consider the problem of determining the winner in various models of reachability games played on counter systems. A game on a graph is played by two players, who own states. During a play, the players create a run on the system by moving a token between states along transitions, the choice of the transition being made by the owner of the state where the token is. When the game is played on a counter system of dimension d, transitions update a vector of d counters, usually by componentwise additions. Configurations in a counter system are pairs composed of a state of the system and a counter vector. In reachability games on counter systems, the objective of the first player is to reach a particular configuration. The decision problem associated with a reachability game is whether the first player has a winning strategy for the game from a given initial configuration, in other words whether the initial configuration is winning. When the problem is decidable, we wonder whether it is possible to describe the set of winning initial configurations.

We study various models of counter reachability games and the effect of differences between models on decidability or complexity of the problems that we consider. The main part of our study follows two directions: on the one hand, we compare models that differ regarding their semantics, and on the other hand we consider a stateless model of counter systems. About the first direction, the semantics of our systems differ regarding what happens when a counter should become negative. We focus primarily on three semantics, the first two being standard: In vector addition systems with states (VASS), or equivalently under the "VASS semantics", a transition that should make a counter become negative is deactivated, so all counters are nonnegative integers all along the play; the semantics that we call "$\mathbb{Z}$ semantics" allows any integer value for all counters; a third semantics, the "non-blocking VASS semantics", consists of letting all transitions active, yet replacing at each step any negative value by zero. About the second direction, we study a model called "robot games", where the system has exactly one state per player and there are no self-loops. In some sense, we abstract from the state in order to focus on the counters and on their evolution. Accordingly, in dimension one, studying robot games involves various results about modular arithmetic, on which we base most strategies that we build. Our motivation for robot games is to have a model of counter reachability games where determining the winner is easier than in the general case. In one of our other models, we allow updates that are resets of the counters along the transitions, and more generally updates defined by componentwise affine functions. We also study systems, named hierarchical counter systems, on which any update of the i^{th} counter requires that the value of all counters from 1 to $i-1$ is zero. Such a restriction is added to the semantics and gives three new semantics.

Our main results regarding semantics are mutual reductions between the decision problems under our three main semantics. Hence, we extend complexity or undecidability results of [BJK10], where games are played on VASS, to other semantics. Under the non-blocking VASS semantics, a particular case has a lower complexity in dimension one: when the counter value in the objective is 0. Determining the winner is then P-complete with a unary encoding instead of PSPACE-complete with a unary encoding when the counter value is a positive integer. Our main results on the model of robot games, are the following. For dimension one, we describe an algorithm with an optimal complexity, according to a reduction that we make from the problem of determining the winner of a countdown game, a model of counter reachability games presented in [JLS07]. As a consequence, in dimension one, determining the winner of a robot game has a lower complexity than determining the winner of reachability games on counter systems. We also prove that robot games are undecidable in dimension three, and we leave the case of dimension two as an open problem. For dimension two, we have written a graphical implementation of a stepwise construction of the winning set to get an insight of the shape that this winning set may have. On the other models that we study, we prove that: (i) allowing counter resets does not affect the complexity of the decision problem in dimension one, (ii) allowing affine updates makes the decision problem undecidable in dimension two, even in the one-player case, and (iii) the decision problem on hierarchical counter systems is in NP for any dimension in the one-player case.

Résumé

Dans cette thèse, nous considérons le problème consistant à déterminer le vainqueur de différents modèles de jeux d'accessibilité sur des systèmes à compteurs. Un jeu sur un graphe est joué par deux joueurs à qui les états du système appartiennent. Au cours d'une partie, les joueurs construisent une exécution du système en déplaçant un jeton d'état en état le long des transitions, le choix de chaque transition étant laissé au joueur à qui appartient l'état où le jeton se trouve. Lorsque le jeu est joué sur un système à compteurs de dimension d, les transitions mettent à jour un vecteur de d compteurs, généralement par l'addition d'un vecteur. Les configurations des systèmes à compteurs sont des paires composées d'un état du système et d'un vecteur formé par les valeurs des compteurs. Dans les jeux d'accessibilité sur des systèmes à compteurs, l'objectif du premier joueur est d'atteindre une configuration particulière. Le problème de décision associé à un jeu d'accessibilité consiste à déterminer si le premier joueur a une stratégie gagnante dans le jeu depuis une configuration initiale donnée, ce qui revient à dire que la configuration initiale est gagnante. Lorsque ce problème est décidable, on s'intéresse à la possibilité de décrire l'ensemble des configurations gagnantes.

Nous étudions divers modèles de jeux d'accessibilité sur des systèmes à compteurs, ainsi que l'effet des différences entre les modèles sur la décidabilité ou la complexité des problèmes que nous considérons. L'essentiel de notre étude suit deux directions : d'une part, nous comparons des modèles dont la différence réside dans la sémantique, d'autre part nous considérons un modèle sans états de systèmes à compteurs. Concernant la première direction, la sémantique dépend du comportement du système quand un compteur devrait devenir négatif. Nous nous focalisons principalement sur trois sémantiques : dans les systèmes d'addition de vecteurs avec états (*VASS* en anglais), une transition qui devrait rendre un compteur négatif est désactivée, de sorte que tous les compteurs sont toujours des entiers naturels ; la sémantique que nous appelons sommairement « $\mathbb{Z}$ semantics » en anglais autorise toutes les valeurs entières pour les compteurs ; à ces deux sémantiques habituelles s'ajoute une troisième, que l'on pourrait traduire par « sémantique de VASS non-bloquant », et qui consiste à autoriser toutes les transitions, en remplaçant d'éventuelles valeurs négatives par zéro. Concernant la seconde direction, nous étudions un modèle intitulé « robot games », où il n'y a qu'un état pour chaque joueur, sans boucle sur les états. Il s'agit en quelque sorte d'ignorer l'état pour ne se concentrer que sur les compteurs et leur évolution. Ainsi, en dimension une, l'étude des robot games fait appel à des résultats d'arithmétique modulaire, sur lesquels nous nous appuyons pour décrire la plupart des stratégies construites. L'étude des robot games est motivée par la volonté d'avoir un modèle de jeux d'accessibilité avec des compteurs pour lequel déterminer le vainqueur est plus facile que dans le cas général. Dans un autre de nos modèles, nous autorisons des mises à

jour qui réinitialisent des compteurs, voire plus généralement des mises à jour qui sont dans chaque dimension des fonctions affines. Nous étudions aussi des systèmes, appelés systèmes à compteurs hiérarchisés, où toute mise à jour du i-ième compteur nécessite que tous les compteurs numérotés de 1 à $i-1$ soient nuls. Une telle restriction est greffée sur la sémantique, ce qui donne en fait trois nouvelles sémantiques possibles.

Nos résultats principaux concernant les sémantiques des systèmes sont des réductions mutuelles entre les problèmes de décision avec les trois sémantiques principales. Ainsi, nous étendons à d'autres sémantiques les résultats de complexité et d'indécidabilité de [BJK10], où les jeux étudiés sont sur des VASS. Avec la sémantique des VASS non-bloquants, un cas particulier a une complexité moindre en dimension une : quand la valeur dans l'objectif est zéro. Déterminer le vainqueur est alors P-complet avec un encodage en unaire au lieu de PSPACE-complet dans le cas général, également avec un encodage en unaire. Nos résultats principaux sur le modèle des robot games sont les suivants. En dimension une, nous décrivons un algorithme de complexité optimale, d'après une réduction que nous établissons depuis le problème consistant à déterminer le vainqueur d'un autre modèle de jeux d'accessibilité : les « countdown games » [JLS07]. Par conséquent, en dimension une, déterminer le vainqueur de robot games a une complexité moindre que déterminer le vainqueur de jeux d'accessibilité sur des systèmes à compteurs. Nous prouvons également que les robot games sont indécidables en dimension trois, et la décidabilité en dimension deux demeure une question ouverte. En dimension deux, nous avons écrit une implémentation graphique de la construction pas à pas de l'ensemble des configurations gagnantes pour avoir une idée de la forme de cet ensemble gagnant. Dans les autres modèles que nous étudions, nous prouvons que : (i) l'ajout de réinitialisations de compteurs ne modifie pas la complexité des problèmes de décision considérés en dimension une, (ii) autoriser des mises à jour affines rend les problèmes de décision indécidables en dimension deux, même avec un seul joueur, et (iii) avec un seul joueur, déterminer si une configuration est accessible sur un système à compteur hiérarchisé est dans NP quelle que soit la dimension.

Chapter 1

Introduction

"Display twenty-one matches in a row. Let two players remove turn after turn between one and three matches each time. The player who picks the last match wins." In three sentences, we sum up one of the most popular versions of Nim games. This makes the first step towards the domain of games with counters, with an example that bears a long history: The general game itself has been solved more than hundred years ago, and a machine that plays it perfectly already existed before the fourties, according to [Eps12].

Counters, which are usually integers due to the way computers store numbers, express anything that we want to quantify, when values or fluctuations are important. For example, if we have too little of a resource, we buy or produce supplies until a critical level, which can be much higher, is reached. The consumption and the refill of the resources itself can be controlled by other agents, viewed as opponents in a game. Other agents can be the environment, with or without a predictable behaviour. Consider for example an automatic air conditioning that maintains the inside temperature in a certain range. The outside temperature has an influence on the inside temperature, hence on the activity of the heater, which will consume more or less electrical power. When we model such a system, we can discretize temperature and energy consumption, with an arbitrary precision, for example hundredths of degrees and watt-hour, in order to work with integers. Where is the game? The programmer of the heater would like to minimize the power consumption such that the temperature requirements remain satisfied. Intuitively, when it is cold outside, it is less expensive to keep the inside temperature in the lower part of the required range, and conversely when it is warm outside.

On the example with the air conditioning, we face the following question: what can a system do with the counters? The previous situation is usually modelled as a timed system, and we can imagine that the increase of the inside temperature is the image of the temperature by an affine function, hence we need to solve a differential equation to obtain this temperature. When we abstract from the time parameter and discretize temperature, we get step by step updates of the counters, and these updates can remain as simple as adding constants. The choice of the constants

to add need not be the same at each step, though. A parameter on which the choice depends can be an internal state of the system, or even a more global state for the whole model, a state that players, so our agents, share and that changes when they play.

Scope of the thesis

In the domain of model checking, the use of integer counters has come naturally to represent any discrete parameter that one wants to control or measure. This has led to various models of counter systems, according to the phenomenon that counters represent. Whenever an uncontrollable antagonist, human or not, takes part in the phenomenon, counter systems become game arenas. For example, when the system is a device that must perform a given task without any further human intervention and the antagonist is the environment, the program of the device must deal with any possible case, in particular the least favorable case, that can happen in the environment. We represent such a program as a player that uses an optimal strategy against the strategy of his adversary.

Since the introduction of counter machines by Minsky in [Min67], the number of types of special Turing machines with integers on which they perform simple operations increased, according to features that were needed. Usually, with few different operations already, the machines were Turing-powerful with at least two counters, the key operation being the ability to test that a counter is zero. Petri nets, introduced in [Pet62], and Vector Addition Systems (VAS), introduced in [KM69], are famous equivalent models of counter machines: without zero tests, in fact with positivity tests instead, the problem of reaching a particuliar configuration is decidable for these models, according to [May84]. In our work, we take as a basis the semantics of Vector Addition Systems with States (VASS), which are also equivalent to VAS, a possible reduction being given in [HP79]. VASS become game arenas by assigning each vertex to one of the two players, Adam and Eve. The games that we consider are always turn-based. Actually, it stands as an exception to study concurrent games on counter systems, which is the case in [BHSS12]. In [BJK10], the authors give decidability and complexity results for the problem of determining the winner of reachability games on VASS, where the objective is to make at least one counter zero. In [Cha10], the complexity of the same problem is improved when the number of counters is two. For the particular objective of making at least one counter zero, it turns out that the problem of determining the winner is decidable in any dimension, which is seldomly the case in the models that we study, the difference being that the objectives are singletons.

However, the literature also contains examples where the systems do not have the restriction that counters remain positive like in VASS. In our work, we call $\mathbb{Z}$ semantics the semantics under which counters can take any integer value. It is then conceivable to define as an objective that counters remain positive. This is the case in energy games [BFL$^+$08, FJLS11, CD10], which are linked to mean-payoff games [CDHR10, CD10, CD11] and in which the objective of one of the players is that all counters remain nonnegative. Another model, consumption games, presented in [BCKN12], is like energy games but with ω-transitions, that add an arbitrarily large integer to one of the counters.

In the games that we study, we only consider reachability objectives. In some sense, reachability objectives are very standard, as they are just expressed by the formula "at some step, the configuration is in a particular set", and the particular set is usually a singleton. Note that an energy game can be reduced to a reachability game, although the most immediate reduction requires to add two vertices linked with $-k$ edges, where k is the lowest integer that is added while crossing an edge, and k is necessarily negative if the game is nontrivial. Also, in our counter systems, a player has a strategy to prevent the counters from becoming unbounded if, and only if, there exists a configuration c such that the same player has a strategy to reach c from the initial configuration and from c itself. Detection of controllable zero-cycles is involved, and also doable in polynomial time in the one-player case, as stated by the authors of [KS88].

Regarding the labels of the edges when counter updates are vector additions, it is important in terms of complexity to make a difference between the models where the labels are vectors with only -1, 0 or 1 as components, and the models where labels can be any integer vector. In some sense, this distinction is the difference between binary and unary encoding. One of our reductions, namely between two decision problems for reachability games with singleton and non-singleton objectives, is polynomial for unary encodings but not for binary encodings. Actually, some authors, like in [BJK10] and [Cha10], even rule out the case where other integers than -1, 0 and 1 appear in the label of edges.

In our work, some models that we present are less common than VASS or counter systems without restriction. This is for example the case of systems, which we call non-blocking VASS, where negative counter values are replaced by zero. On these systems, the usual decision problems have a lower complexity in dimension one than similar problems on VASS. Another uncommon model introduces a hierarchy between the counters, and the semantics of the model requires that a counter cannot be modified if a lower counter is not zero. The latter model is inspired by VAS with hierarchical zero-tests, presented in [Rei95]: we attach states and a hierarchical zero-test, possibly void, at every step. VAS with hierarchical zero-tests extend VAS with one zero-test, on which some decision problems, although undecidable with arbitrary zero-tests, become decidable, as we can see in [AM09, BFLZ10, BFLZ12, Bon11, FS10, Rei08].

Outline

In Chapter two, we present reachability games on counter systems under three different semantics. In the first section, we present the semantics of vector addition systems with states, or VASS semantics. Because the VASS semantics is the most common semantics, we take the problem of determining the winner under it as the basis for our reductions to give results under other semantics.

The second section deals with the $\mathbb{Z}$ semantics, i.e., the semantics under which counter values may be negative. We begin with the one-player case, for which the reachability problem has small complexity compared to the reachability problem on VASS. After that, we prove that the problem of determining the winner in reachability games of dimension two, which is undecidable under the VASS semantics, is also undecidable under the $\mathbb{Z}$ semantics. We end the section by showing that the problem of determining the winner of reachability games has the same complexity in dimension

one under the $\mathbb{Z}$ semantics and under the VASS semantics.

The third section introduces the non-blocking VASS semantics. In our study of counter reachability games, we distinguish whether the counter value in the objective is positive or zero. In the first case, the associated decision problems for VASS and non-blocking VASS are interreducible. In the second case, the associated decision problem has a much lower complexity for non-blocking VASS. The first three sections form an extension of the work presented in [Rei13].

The fourth section presents an extended model, where counter updates include resets to zero. We prove that the problem of determining the winner of a one-dimensional counter reachability game with resets is decidable with the same complexity as in the model without resets. In our proof, we consider systems under the $\mathbb{Z}$ semantics, yet the reduction that we give also holds under the other two semantics.

In the fifth section, we study a generalization that subsumes counter additions and resets: affine updates. Here, when an edge is taken, the value of each counter is modified according to the affine function in the corresponding component of the label of the edge. We prove that the reachability problem, so a one-player game, is undecidable in dimension two.

In the sixth section, we give three new semantics, which are restrictions of the $\mathbb{Z}$, VASS and non-blocking VASS semantics. Under the new semantics, which we call semantics of hierarchical counter systems, for any i between 1 and the dimension, the i^{th} counter can only be modified if counters 1 to $i-1$ are zero. Whether the counters can become negative or not depends on the combination with the three main semantics. Here again, we just consider the reachability problem and we prove that it is in NP for any dimension and under any semantics.

In Chapter three, we introduce robot games, a particular case of counter reachability games under the $\mathbb{Z}$ semantics. In a robot game, each player owns exactly one vertex, and there are no self-loops, so plays are round-based. Also, Adam always starts and Eve's objective is to reach the origin after one of her turns. Robot games have many regularity properties, for example Eve's winning set is closed under addition. Also, in some cases, a human can solve the game, especially in dimension one: When all moves are even, no odd position is winning for Eve; When Adam has at least one positive move and Eve has no negative moves, no positive position is winning for Eve; etc.

In the first section, we write an EXPTIME algorithm that computes Eve's winning set in dimension one as a union of a finite set, around zero, and a void or infinite set, beyond given bounds towards $\pm\infty$. We prove the correctness of the algorithm using a deeper view of the properties of robot games than what we present earlier in this paragraph.

In the second section, we give a lower bound for the problem of determining the winner of a robot game in dimension one, namely EXPTIME-hard, which makes our algorithm asymptotically optimal. The proof is based on a reduction of countdown games, introduced in [JLS07], which are seen as a particular model of counter reachability games. The first two sections are based on the work presented in [AR13].

In the third section, we show that the problem of determining the winner of a robot game is undecidable in dimension three, with a technical proof that we present in three steps.

In Chapter four, we give some comments about robot games in dimension two, based on a graphical tool that implements a step-by-step construction of the winning set of Eve. With the concrete results and the examples of instances that give a winning set of uncommon shape, we have arguments for and against decidability of the problem of determining the winner, which we leave as open.

Contents

Chapter 2

Reachability Games on Counter Systems

Consider an integer variable x. Give somebody a list of instructions to modify the value of x. We can wonder whether x remains positive, or whether x takes at least once, or infinitely often, a given value, or, on the contrary, whether x grows unboundedly at some point...

All these questions are conceivable winning conditions for one of the players in a game on a counter system. Among the conditions, reachability, i.e., whether the variable takes at least once a given value, plays a central role because many other conditions reduce to reachability: for example, a counter remains positive if it does not reach any negative value.

In this chapter, we introduce reachability games on counter systems, starting with the well-known model of *Vector Addition Systems with States* (VASS). We study three semantics, which represent three different behaviours when an update may decrease a counter below zero: crossing an edge with such an update is either impossible, which is the *VASS semantics*, or possible, which is the $\mathbb{Z}$ *semantics*, or possible but negative values are replaced by zero, which is the *non-blocking VASS semantics*. The latter semantics, which we studied in [Rei13], is new, as far as we know.

The objectives that we consider are usually configurations, i.e., a given counter vector must be reached on a given vertex. We show that determining the winner of a reachability game in dimension one is PSPACE-complete under the three studied semantics when the counter value in the objective is in unary. Surprisingly, there is a particular case : under the non-blocking VASS semantics, when the objective is a configuration with counter value 0, determining the winner is P-complete, because the winning set for Eve is downward closed, a property that is specific to the objective under the non-blocking VASS semantics. In dimension two, determining the winner is undecidable under the VASS semantics when the objective is a configuration, according to [BJK10], and we give a reduction to the same decision problem, which we hence prove to be undecidable, under the $\mathbb{Z}$ semantics.

The last result contrasts with the one-player case, for which the reachability problem is NP-complete under the $\mathbb{Z}$ semantics, but solely proved decidable by the author of [Kos82], without any known lower bound when the dimension is at least three, under the VASS semantics.

We extend the results of this chapter with three other models of counter systems. The first model has an ordinary feature: reset edges. Even though translation invariance, a property of reachability games under the $\mathbb{Z}$ semantics, disappears, the complexity of determining the winner in dimension one is the same as the complexity without resets. The second model has affine updates of the counters, a restricted case of the polynomial updates of [FGH13]. In the third model, some edges are disabled depending on the counter values, like in VASS but under another condition: edges that modify the i^{th} counter can be taken only if the first $i-1$ counters are zero.

Definitions

A *counter system* (Figure 2.1) in dimension d is a directed graph (Q, T), where Q is a finite set of *vertices* and $T \subseteq Q \times \mathbb{Z}^d \times Q$ is a finite set of *edges* labelled by integer vectors. A *path* in a counter system (Q, T) is an infinite sequence of vertices that starts from an initial vertex $q_0 \in Q$ and such that, for every $i \in \mathbb{N}$, there exists a vector v_i such that $(q_i, v_i, q_{i+1}) \in E$. A *configuration* in a counter system is a pair (q, x), where $q \in Q$ and $x \in \mathbb{Z}^d$. A *run* of a counter system (Q, T) is an infinite sequence $r = (q_0, x_0)(q_1, x_1)\dots$ that starts from an initial configuration $(q_0, x_0) \in Q \times \mathbb{Z}^d$ and such that, for every $i \in \mathbb{N}$, we have $(q_i, x_{i+1} - x_i, q_{i+1}) \in E$. Hence, a run of a counter system follows a path. A counter system is *unary* if all vectors in the labels of the edges are in $\{-1, 0, 1\}^d$.

As we define them, the counter updates of the counter systems that we study here are vector additions. In the literature, counter systems may have affine updates, for example in [FGH13], there may be resets or transfers between counters, which is the case in [DFS98], and the counter values may even decrease arbitrarily, to describe situations where the system is lossy, like in [BM99].

A *counter reachability game* (Figure 2.2) is played by two players, Eve and Adam, on a counter system (Q, T). We partition the set of vertices into $Q_E \uplus Q_A$; Eve owns Q_E, and Adam owns Q_A. In our figures, we use $\bigcirc$ to represent Eve's vertices, $\square$ to represent Adam's vertices and $\diamond$ when the owner of the vertex does not matter.

A *play* in a counter reachability game corresponds to a run of the counter system. Hence, a play is represented by an infinite sequence of configurations that players form by moving a token on (Q, T) and updating counters as follows. At the beginning, the token is at a vertex q_0 and the counter vector is initialized with x_0, hence the initial configuration is (q_0, x_0). If the token is at $q \in Q_E$, then Eve chooses an edge (q, v, r), otherwise Adam chooses. The token is moved to r, the counter vector, which was x before, is updated to $x + v$, and the configuration $(r, x + v)$ is appended to the play. There is a special configuration, called the *objective* of the game. Eve wins if the play visits the objective, and Adam wins otherwise, for example when a dead end is reached. Note that our definition of objective is not the usual definition, because objectives need not be a single configuration in general.

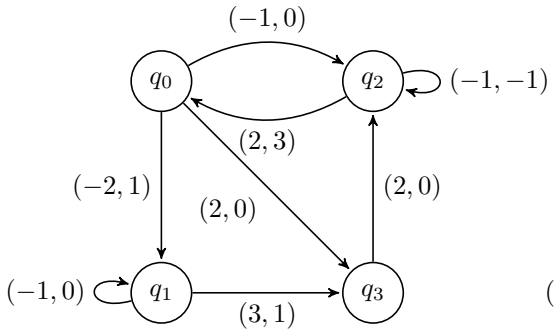

Figure 2.1 – Two-dimensional counter system (Q,T).

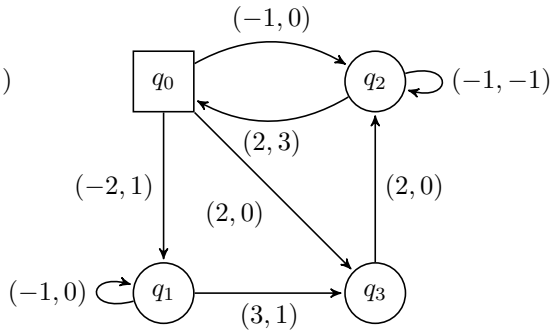

Figure 2.2 – Counter reachability game on (Q,T).

In our example of Figure 2.2, let $(q_0,(2,4))$ be the objective of the game. A possible play starting in q_0 with $(1,1)$ as initial counter is the sequence that starts with

$$(q_0,(1,1))(q_3,(3,1))(q_2,(5,1))(q_2,(4,0))(q_0,(6,3))$$

and that alternates between q_0 and q_2. The play that we have given cannot visit the objective, because the difference between the second and the first counter only increases upon the loop. However, if the first move of Adam would have been $(q_0,(1,1)) \to (q_2,(0,1))$, then Eve would have won by choosing the edge to q_0, and it would have been useless to look at the rest of the play.

A *play prefix* starting from the configuration (q_0,x_0) in a counter reachability game is a finite sequence $(q_0,x_0)(q_1,x_1)\dots(q_k,x_k)$ of configurations in the counter system (Q,T). A *strategy* for a player is a function that takes as argument a play prefix $(q_0,x_0)(q_1,x_1)\dots(q_k,x_k)$ and returns an edge that is enabled from the configuration (q_k,x_k), depending on the semantics. Given a configuration (q_0,x_0), two strategies s_E and s_A for the players, the *outcome* of s_E and s_A from the configuration is the play starting at (q_0,x_0) and obtained when each player always chooses edges according to his strategy. A strategy s is *winning* for a player P from a given configuration c if, for any strategy s' of the other player, the outcome of s and s' from the configuration c is a play that player P wins. A configuration (q_0,x_0) in the game is *winning* if Eve has a winning strategy from (q_0,x_0). The decision problem associated with a counter reachability game is, given a configuration c of a counter system under a semantics that is defined beforehand, to determine whether Eve has a winning strategy from the configuration c.

Again in our example, Adam has a winning strategy from the configuration (q_0,x_0) for any vector $x_0=(y_0,z_0)$, provided that the objective (q_f,x_f), with $x_f=(y_f,z_f)$, is different from the configuration (q_0,x_0). A possible winning strategy involves comparing the differences between the two counters in the objective and in the starting configuration. If z_f-y_f is greater than or equal to z_0-y_0, then Adam always chooses the edge to q_3, and we notice that the difference between the second and the first counter never increases. Otherwise, Adam always chooses the edge to q_2 and

the difference between the second and the first counter never decreases. In any case, Eve has no possibility to reach the objective.

We restrict to a unique configuration in our reachability objectives, which allows to encode most kinds of reachability conditions, as stated in Proposition 12 on page 34. Actually, a consequence of this proposition is that, when the complexity of determining the winner is above EXPTIME or when the system is not unary, by a slight modification of the arena, we can replace by a singleton any objective that is a union of $\{q_f\} \times X_f$, where $q_f \in Q$ and $X_f \subseteq \mathbb{Z}^d$ is linear, i.e. has the form $\{x_0 + k_1 x_1 + \cdots + k_n x_n \mid k_1, \ldots, k_n \in \mathbb{N}\}$ for some d-dimensional vectors $x_0, x_1, \ldots, x_n$. When the complexity of determining the winner is below EXPTIME on unary counter system, this complexity may differ depending on whether the objectives are singletons or linear sets: determining whether Eve has a winning strategy to reach for example the value 0 is P-complete if 0 can be reached in any vertex but PSPACE-complete if 0 must be reached in a given vertex. This property is proved in [Ser06] (membership) and [BJK10] (hardness) for VASS, and adapted for systems under the $\mathbb{Z}$ semantics.

We introduce a notation for the decision problems with which we deal, and we write $\mathit{semantics}^1_d(x_f)$ with the following parameters:

- the semantics : Vass, Cs for the $\mathbb{Z}$ semantics to avoid notation confusions, or NBVass for non-blocking VASS;
- a subscript d for the dimension;
- an optional argument x_f for the counter value in the objective;
- a superscript 1 to point out, if present, that the system is unary.

We omit the vertex in the objective, because only the counter value is relevant with regard to complexity. For example, let us look at two notations that appear in this chapter.

- The problem of determining the winner on a counter system of dimension two with an arbitrary objective is denoted by Cs_2.
- The problem of determining the winner on a unary VASS of dimension one with 0 as objective value is denoted by $\text{Vass}^1_1(0)$.

A *linear set* in $\mathbb{Z}^d$ is a set of the form $\{x + \sum_{i=1}^n k_i x_i \mid k_1, \ldots, k_n \in \mathbb{N}\}$, for some d-dimensional vectors $x, x_1, \ldots, x_n$. The set $\{x + \sum_{i=1}^n k_i x_i \mid k_1, \ldots, k_n \in \mathbb{N}\}$ is the least set that contains x and is closed under addition of vectors in $\{x_1, \ldots, x_n\}$. We denote this linear set by $x + \langle\{x_1, \ldots, x_n\}\rangle_{\mathbb{N}}$ or simply $x + x_1\mathbb{N}$ when $n = 1$. We also write $\langle Y\rangle_{\mathbb{N}}$ rather than $0 + \langle Y\rangle_{\mathbb{N}}$. We say that a vector is *Y-reachable* if, and only if, it belongs to $\langle Y\rangle_{\mathbb{N}}$. When $d = 1$, we have the notion of Y-reachable integers.

In our work, we write 0^d for the d-dimensional vector that has 0 in all its components, and we denote by $-\mathbb{N}$ the set $\{0, -1, -2, \ldots\}$ of nonpositive integers.

2.1 Reachability games on Vector Addition Systems with States

Reachability games on counter systems, as well as the reachability problem, appear most often when the counter system is a Vector Addition System with States. The article [BJK10] covers many results about games on unary VASS and on unary extended VASS. Extended VASS are VASS with possible additional edges that add ω to a counter, which means adding an arbitrarily large nonnegative integer. Like in our definitions, states, i.e. vertices, belong to either player.

We rephrase the first result of the article to fit our notations.

Theorem 1 ([BJK10], Proposition 4)**:** *Let (Q,T) be a unary* VASS *of dimension two. Consider a reachability game on (Q,T) with $Q_Z \times (\{0\} \times \mathbb{N}) \cup (\mathbb{N} \times \{0\}))$ as objective, where $Q_Z \subseteq Q$. The problem of determining the winner of this game is undecidable.*

The following proposition holds under all sematics and makes most proofs of this chapter simpler, without loss of generality.

Proposition 2: *Let (Q,T) be a counter system. Consider a reachability game on (Q,T) with a union of $\{q_i\} \times X_i$ as objective, for some vertices q_i and some subsets X_i of $\mathbb{Z}$. We can build a counter system (Q',T') on which we consider a reachability game with a union of $\{q'_i\} \times X_i$ as objective, such that all q'_i are vertices of Eve, and she has a winning strategy on (Q,T) if, and only if, she has a winning strategy on (Q',T').*

Proof. Let (Q,T) be a counter system. Let $Q_Z = \{q_1, \dots, q_n\}$ be a subset of Q such that the objective for a reachability game on (Q,T) is $\{\{q_i\} \times X_i \mid 1 \leq i \leq n\}$ for some subsets $X_1, \dots, X_n$ of $\mathbb{Z}$.

We build the counter system (Q',T') such that Q' is the union of Q and of one copy of each vertex of Adam in Q_Z. If q_i is a vertex of Adam in Q_Z, which we call duplicated vertex, we denote by q'_i the copy that we create. The set T' is the union of the set of all edges that do not point to a duplicated vertex and of the set $\{(q,v,q'_i),(q'_i,0^d,q_i) \mid (q,v,q_i) \in T\}$. In other words, any edge to a duplicated vertex is split into two edges, the first edge updates the counters and points to a new vertex of Eve and the second edge, the only outgoing edge of the new vertex, points to the duplicated vertex.

We set the objective to be the union of the set of the $\{q'_i\} \times X_i$, where q_i is a duplicated vertex, and of the set of the $\{q_i\} \times X_i$, where q_i is a vertex of Eve. There is no difference in how the two games are played, so Eve has a winning strategy in the second game if, and only if, she has a winning strategy in the first game, because on (Q',T') the counter value is already updated like in (Q,T) when a copy of a duplicated vertex is reached. □

While the objective in Theorem 1 is a set of configurations, our model restricts to singletons. Still, we show that both are equivalent in any dimension.

Proposition 3: *Let (Q,T) be a* VASS *of dimension d. Consider a reachability game on (Q,T) with $Q_Z \times (\{0\} \times \mathbb{N}^{d-1}) \cup (\mathbb{N} \times \{0\} \times \mathbb{N}^{d-2}) \cup \cdots \cup (\mathbb{N}^{d-1} \times \{0\})$ as objective, where $Q_Z \subseteq Q$. We can build a* VASS *(Q',T') such that Eve wins the reachability game on (Q,T) if, and only if, she wins the reachability game on (Q',T') with objective $(\bot, 0^d)$, where $\bot \in Q' \setminus Q$.*

Proof. According to Proposition 2, we assume that Q_Z contains vertices of Eve only. We build a VASS (Q',T'), where $Q' = Q \cup \{\emptyset_1, \emptyset_2, \ldots, \emptyset_d, \bot\}$, and

$$\begin{aligned} T' = T &\cup \{(q, 0^d, \emptyset_1), (q, 0^d, \emptyset_2), \ldots, (q, 0^d, \emptyset_d) \mid q \in Q_Z\} \\ &\cup \{(\emptyset_1, (0,-1,0\ldots,0), \emptyset_1), (\emptyset_1, (0,0,-1,0,\ldots,0), \emptyset_1), \ldots, (\emptyset_1, (0,\ldots,0,-1), \emptyset_1)\} \\ &\cup \{(\emptyset_2, (-1,0,0\ldots,0), \emptyset_2), (\emptyset_2, (0,0,-1,0,\ldots,0), \emptyset_2), \ldots, (\emptyset_2, (0,\ldots,0,-1), \emptyset_2)\} \\ &\ldots \\ &\cup \{(\emptyset_d, (-1,0\ldots,0), \emptyset_d), (\emptyset_d, (0,-1,0,\ldots,0), \emptyset_d), \ldots, (\emptyset_d, (0,\ldots,0,-1,0), \emptyset_d)\} \\ &\cup \{(\emptyset_1, 0^d, \bot), (\emptyset_2, 0^d, \bot), \ldots, (\emptyset_d, 0^d, \bot), (\bot, (0,0), \bot)\}. \end{aligned}$$

Note that (Q',T') is unary if (Q,T) itself is unary. If Eve has a winning strategy in the game on (Q,T), then she can follow the same strategy on (Q',T') and reach a configuration where the vertex is in Q_Z and one of the counters is zero. At this point, she can go to the vertex where she decreases the other counters to zero and, after that, she can go to $\bot$ and win. Conversely, if Eve has a winning strategy in the game on (Q',T'), then she can enforce that the play visits $Q_Z \times (\{0\} \times \mathbb{N}^{d-1}) \cup (\mathbb{N} \times \{0\} \times \mathbb{N}^{d-2}) \cup \cdots \cup (\mathbb{N}^{d-1} \times \{0\})$, as this is the only possibility to reach a vertex $\emptyset_i$ with the i^{th} counter at zero and, after that, to reach the objective. □

The article [BJK10] also considers a game on their extended VASS, where the system wins if, and only if, a configuration with at least one counter at 0 is reached, so the vertex does not matter anymore. The problem of determining the winner of this game is in $(d-1)$-EXPTIME, where $d \geq 2$ is the dimension. In dimension one, the same problem is in P, and PSPACE-complete when the objective is $Q_Z \times \{0\}$, with $Q_Z \subseteq Q$. The PSPACE upper bound is a consequence of a theorem in [Ser06], where the author studies games on one-stack pushdown systems, from which results on unary VASS are inherited. In dimension two, the complexity of determining the winner is actually P instead of EXPTIME, an improvement given by the author of [Cha10].

We have already mentioned the difference between the objective in reachability games on VASS that the authors of [BJK10] study and our model: the objective is on the one hand "at least one counter is zero" and on the other hand "all counters are zero". Because both are equivalent in dimension one, we may use the results directly.

Proposition 4: *Determining the winner of a reachability game on a one-dimensional VASS where Eve wins if, and only if, the counter value becomes* 0 *reduces in polynomial time to* $\textsc{Vass}_1(0)$*, and vice-versa.*

The difference between the two objectives in this proposition is that with the first objective Eve wins when the counter value is 0, whichever is the vertex, whereas with the second objective Eve wins when the counter value is 0 in a given vertex.

Proof. The reduction from the problem of reaching the value 0 to $\textsc{Vass}_1(0)$ is straightforward: it suffices to give to Eve the possibility of going to a new sink $\perp$ after any edge, with two-state gadgets that replace Adam's vertices. The idea behind the gadgets is like in Proposition 2.

Let us prove the reduction from $\textsc{Vass}_1(0)$ to the problem of reaching the value 0. Let (Q, T) be a VASS of dimension one. Consider a reachability game on (Q, T), where Eve wins if, and only if, the counter value becomes 0 in a particular vertex q_f, which belongs to Eve. Suppose that Q is partitioned into Q_A and Q_E, which are the sets of vertices of Adam and Eve, respectively.

Let $Q' = Q \cup \{\perp\}$, where Adam owns Q_A and Eve all other vertices. The set T' contains the edge $(q, 2v, r)$ for all edges (q, v, r) of T. There are also two new edges: $(q_f, -1, \perp)$ and $(\perp, 0, \perp)$.

Intuitively, a play on (Q', T') corresponds to a play on (Q, T), but with the possibility for Eve to force to reach the sink $\perp$ whenever the configuration is $(q_f, 0)$ in (Q, T). Hence, if the play on (Q, T) starts in (q_0, x_0), then the corresponding play on (Q', T') starts in $(q_0, 2x_0 + 1)$. The counter value in (Q', T') is always one plus twice the counter value in (Q, T), until the sink $\perp$ is reached. At this point, the counter value in (Q', T') is 0 if, and only if, it is zero in (Q, T), hence Eve has a winning strategy in one game if, and only if, she has a winning strategy in the other. □

A remark about the reduction from $\textsc{Vass}_1(0)$ to the problem of reaching the value 0 and the reverse reduction: both reductions also hold under the $\mathbb{Z}$ semantics, and for any counter value in the objective (provided that it is doubled in (Q', T')), but the reduction from $\textsc{Vass}_1(0)$ to the problem of reaching the value 0 does not hold under the non-blocking VASS semantics when the counter value in the objective is 0.

Note that for any edge in T that adds an integer k to the counter, there is an edge in T' that adds $2k$ to the counter. Hence, if there is any edge in T that adds another integer than 0 to the counter, then (Q', T') is not unary. As a consequence, the complexity result of [BJK10] cannot be used, and this justifies the difference in the article between the complexities with one or another objective.

To sum up, the problem of determining the winner of a reachability game under the VASS semantics is undecidable in dimension two, and in dimension one, the problem is PSPACE-complete on unary counter systems. Moreover, according to the unpublished short paper [Hun14], the problem of determining the winner of a reachability game on a non-unary system under any semantics is EXPSPACE-complete; membership in EXPSPACE is deducible from the PSPACE-completeness of $\textsc{Vass}_1^1(0)$ by splitting edges of a system to get an at most exponentially bigger unary system. We refer to all these results in the next sections when we give reductions from one of the previous problems.

2.2 Reachability games on counter systems under the $\mathbb{Z}$ semantics

Counter systems under the $\mathbb{Z}$ semantics have a regularity property that is missing under the other semantics. Indeed, if we consider any run on the underlying graph, this run is always possible on the counter system, and its effect on the counters is always the same: along the run, a vector that only depends on the edges that were crossed is added to the initial counter vector.

The first consequence of this property is that we can and will consider the vector 0^d as the only objective, thanks to a simple translation of the initial configuration: determining whether (q_f, x_f) is reachable form (q_0, x_0) and determining whether $(q_f, 0^d)$ is reachable form $(q_0, x_0 - x_f)$ are equivalent. This will not only enable to lower the complexity of decision problems, but also help to make links to other semantics with more freedom.

In this section, we first consider one-player reachability games under the $\mathbb{Z}$ semantics, in other words the reachability problem. Determining whether there is a winning strategy, i.e., a run from the initial configuration to a specific objective, is NP-complete in any dimension, which contrasts to the case of VASS. This difference is a consequence of the regularity property that we mention, insofar as we may consider a run as the multiset of the edges that it crosses, without caring for the order because no edge is ever blocked. Actually, only the fact that a multiset of edges indeed corresponds to a path matters, if the total sum of the labels is the vector that must be added. For VASS, the reachability problem is NP-complete in dimension one, though, as stated in [HKOW09], and, according to the recent article [BFG+14], the reachability problem for VASS is PSPACE-complete in dimension two, which improves the previous lower bound of 2-EXPTIME given in [HRHY86].

We also study reachability games on counter systems under the $\mathbb{Z}$ semantics. Now, the order of the edges matters again, because Adam's strategy takes the current configuration into account, and this configuration depends on the prefix of the play. In fact, we prove that the decision problem is equivalent under the $\mathbb{Z}$ semantics and under the VASS semantics, with mutual reductions that rely on gadgets. These gadgets show how to simulate under one semantics the behaviour around zero of the other, with interventions of a player against the moves of his adversary.

We use the equivalence between the problems to deduce that determining the winner of reachability games under the $\mathbb{Z}$ semantics is undecidable in dimension two and PSPACE-complete in dimension one when the system is unary.

Some results in this section are folklore, with an alternative proof.

2.2.1 The one-player version

The *reachability problem* on counter systems consists of determining whether there exists a run from an initial configuration to a final configuration in a counter system. This problem corresponds

to the one-player version of the decision problem associated with counter reachability games. The reachability problem on VASS has been widely studied. It was proved to be decidable in [Kos82], but with still unknown complexity when the dimension is not fixed, or even fixed but at least three.

The following fact is folklore.

Theorem 5: *The reachability problem on counter systems under the $\mathbb{Z}$ semantics is* NP*-complete.*

Proof. **(NP-membership)** Let (Q,T) be a counter system under the $\mathbb{Z}$ semantics in an arbitrary dimension d. We suppose without loss of generality that there are no self-loops in (Q,T). Let (q_0,x_0) and (q_f,x_f) be two configurations. We consider the problem of determining whether (q_f,x_f) is reachable from (q_0,x_0). Because of the $\mathbb{Z}$ semantics, it is equivalent to decide whether $(q_f,0^d)$ is reachable form (q_0,x_0-x_f), hence we suppose that x_f is the vector 0^d.

When we want to guess a run from (q_0,x_0) to $(q_f,0^d)$, where we also suppose that $q_0\neq q_f$, we just need to guess how many times each edge is taken and check that every vertex is entered as many times as it is exited, apart from q_0 and q_f for which there is a difference of ± 1 time. We also need to check that it is possible to form a path with the number of times each edge is taken, and we explain separately how to do this. We then rewrite the reachability problem as an *integer linear programing* (ILP) problem. Note that solving an ILP is NP-complete, according to [PS82, p. 320, Th. 13.4].

Let n be the size of Q, we assign a number between $d+1$ and $d+n$ to each vertex, where i_0 is the number for q_0 and i_f is the number for q_f. We create vectors in dimension $d+n$ from the vectors in the edges: For each (q,v,q') in T, let i be the number of q and $i'\neq i$ be the number of q', we denote by v' the vector with the same first d components as v, a -1 in the component i_f, an 1 in the component i_0 and a 0 in the other components. We also denote by x_0' the vector with the same first d components as x_0, a -1 in the component i_0, an 1 in the component i_f and a 0 in the other components. We introduce a variable a_i for each edge $(q_i,v_i,q_i')\in E$, and we want to solve

$$\begin{array}{c}\sum_i a_i v_i' = -x_0' \\ \text{under the constraint} \quad \forall i, a_i \geq 0.\end{array}$$

It is equivalent to have a solution for the ILP problem and a run from (q_0,x_0) to $(q_f,0^d)$.

Now, let us show how we check whether a run candidate, i. e., a number of times, possibly 0, each edge is taken, that satisfies the above ILP. We write $n(e)$ the number of times the edge e is taken according to the run candidate that we check. Let us call marked edge an edge e such that $n(e)>0$, and marked vertex a vertex that is the source or the target of a marked edge. Let q be a marked vertex. We need to ensure that there is a path from q_0 to q and a path from q to q_f that crosses only marked edges. Once we have ensured this for all marked vertices, it is guaranteed that there is a path from q_0 to q_f that crosses $n(e)$ times all marked edges e, because we can build a multigraph (Q',E'), where Q' is the set of the marked vertices and E' the set of marked edges e counted $n(e)$ times. The multigrph (Q',E') is connected and there is an Eulerian path from q_0 to q_f in (Q',E') because we have a run candidate (folklore property). The Eulerian path gives a run from (q_0,x_0) to (q_f,x_f).

To sum up, we need to guess a number of times each edge is taken, and for all marked vertices we also need to guess a path of marked edges from q_0 to q_f via the vertex, which can be done in polynomial time as such paths need not visit twice a same vertex.

This polynomial-time reduction proves NP-membership of the reachability problem for counter systems under the $\mathbb{Z}$ semantics.

(NP-hardness) Let us reduce 3SAT to the reachability problem. Consider a formula φ in conjunctive normal form with at most three literals per clause. Let $\{x_1, \ldots, x_n\}$ be the set of variables that appear in φ and $C_1, \ldots, C_k$ be the clauses of φ.

We build an n-dimensional counter system that consists of k gadgets, which represent the clauses, and an end that depends on k and n only. In the gadget, there are one, three or seven edges, if the number of literals is one, two or three, respectively, from the initial vertex to the second vertex. Each of them stands for a satisfying valuation of the variable(s) in the clause: the vector has 1 in the i^{th} component if x_i is set to true in the affectation, -1 if x_i is set to false, 0 if x_i is a variable that does not appear in the clause. After that, there are one vertex and two edges in the gadget for each free variable of the clause, the two edges increment and decrement the component that corresponds to the variable. A gadget for a given formula is depicted in Figure 2.3. At the exit of the last gadget, there is a chain of n vertices with two outgoing edges each, one that adds k to the corresponding component and one that subtracts k from the same component, as we can see in Figure 2.4.

If there is an assignment of the variables that satisfies φ, then there exists a run from the initial vertex of the first gadget with 0^n as counter vector to the end of the chain with 0^n as counter vector, where, in each gadget, each component is incremented when the corresponding variable is assigned to true and decremented else, then after the k gadgets the integer k is subtracted from any component such that the corresponding variable is assigned to true and added to the other components. This is possible because in each gadget each component is either incremented or decremented and the only restriction is the satisfaction of the corresponding clause. Conversely, if there exists a run from the initial vertex of the first gadget with 0^n as counter vector to the end of the chain with 0^n as counter vector, then the vector must be only composed of $\pm k$ at the beginning of the chain, and all gadgets must change the vector identically, which gives an assignment that satisfies the formula.

Note that the number of vertices, as well as the number of edges, is polynomially bounded by the size of the formula. □

In dimension one, it is also possible to reduce the SUBSET-SUM problem, which is proved to be NP-complete in [CLRS09, p.1097], to the reachability problem. This gives another proof of NP-hardness.

Theorem 6: *The reachability problem on counter systems in dimension one under the $\mathbb{Z}$ semantics is* NP*-complete.*

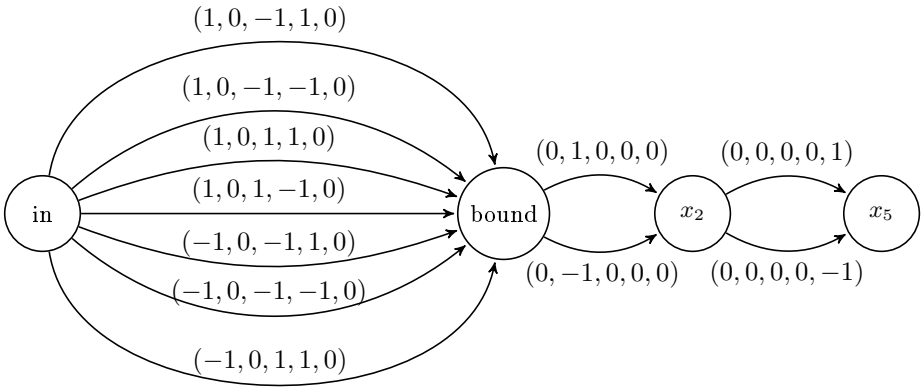

Figure 2.3 – Gadget for the clause $x_1 \vee \neg x_3 \vee x_4$ in a formula with five variables.

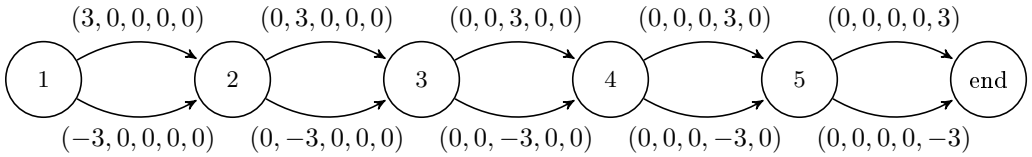

Figure 2.4 – Chain at the end of a counter system obtained from a formula with three clauses and five variables.

We do not give the details of the reduction, which is similar to the one that we use to prove the following fact.

Theorem 7: *The reachability problem on unary counter systems under the $\mathbb{Z}$ semantics is* NP*-complete.*

Proof. NP-membership is a consequence of NP-membership on non-unary systems.

(NP-hardness) Consider an instance (X, s) of Subset-Sum, i.e., a set $X = \{x_1, \ldots, x_n\} \subseteq \mathbb{N}$ and an integer $s \in \mathbb{N}$. Let $d = 1 + \lfloor \log_2(\max(X)) \rfloor$. All elements of X can be written with at most d bits. We build a d-dimensional unary counter system (Q, T) such that the configuration $(q_n, (s, 0, \ldots, 0))$ is reachable from $(q_0, 0^d)$ if, and only if, there exists a subset of X that sums up to s.

We use a binary encoding of the integers of X: We associate a vector $v_i = (v_{i,1}, \ldots, v_{i,d})$ with each x_i, such that $x_i = \sum_{j=1}^{d} v_{i,j} 2^{j-1}$. The set of vertices Q consists of two parts, Q_a to add some vectors v_i to the counter vector, and Q_t to transfer everything to the first dimension. More precisely, $Q_a = \{q_0, q_1, \ldots, q_n\}$ and $Q_t = \{r_2, \ldots, r_d\}$, with the following set of edges:

- $\{(q_{i-1}, v_i, q_i), (q_{i-1}, 0, q_i) \mid 0 < i \leq n\}$ to choose whether to add v_i or not, hence to have x_i in the subset or not;
- $\{(q_n, (0, \ldots, \underbrace{1}_{i}, -1, \ldots, 0), r_i), (r_i, (0, \ldots, \underbrace{1}_{i}, \ldots, 0), q_n) \mid 1 \leq i < d\}$ to transfer one unit of the $i+1^{\text{th}}$ dimension into two units of the i^{th} dimension.

At any step, if the counter vector is $(c_1, \ldots, c_d)$, then the sum of the elements of X that correspond to the vectors that have been added is $\sum_{i=1}^{d} c_i 2^{i-1}$. In q_n, it suffices to transfer everything to the first dimension, where the value should be s when the run simulates a positive instance of Subset-Sum. Figure 2.5 represents an illustration of this reduction. □

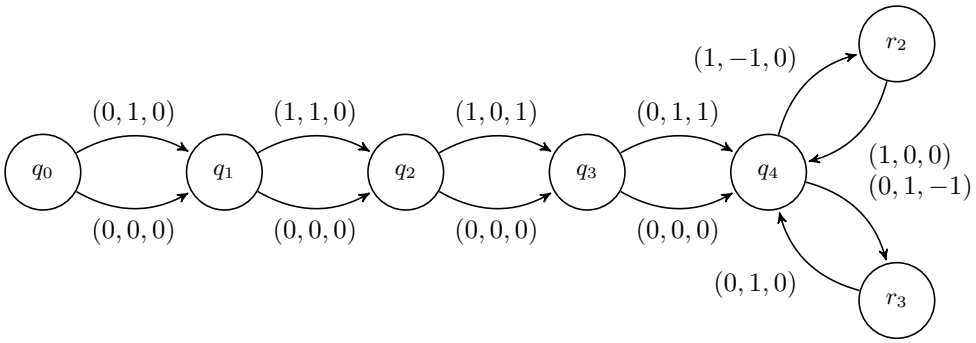

Figure 2.5 – Counter system for the instance $(\{2, 3, 5, 6\}, 11)$ of Subset-Sum.

Another proof of Theorem 7 was given independently in [HH14].

2.2.2 The two-player version in dimension two or more

We present a construction that we use here to obtain our proof of undecidability of counter reachability games in dimension two, and in Subsection 2.2.3 to give lower complexity bounds.

In order to show the reduction from the decision problem under the VASS semantics to the decision problem under the $\mathbb{Z}$ semantics, we simulate in the winning condition the deactivation of edges in VASS, which makes the difference to the $\mathbb{Z}$ semantics.

The only edges that may make a counter value become negative are the ones that add a negative integer in at least one component. In other words, the label of such edges is not in $\mathbb{N}^d$. Other edges do not cause any problem regarding the semantics, which justifies our distinction in the proof of the next proposition.

Proposition 8: $\textsc{Vass}_d(0^d)$ *reduces to* $\textsc{Cs}_d(0^d)$ *in polynomial time for any dimension* d.

Proof. Let (Q,T) be a VASS of dimension d, let (q_0,x_0) and (q_f,x_f) be configurations of (Q,T). We consider the reachability game on (Q,T) where the objective is (q_f,x_f). As usual, the set Q is partitioned into the set Q_E of Eve's vertices and the set Q_A of Adam's vertices. We assume that $q_f \in Q_E$. This is without loss of generality like in the proof of Proposition 3 on page 21.

We build a counter system under the $\mathbb{Z}$ semantics on which Eve has a winning strategy from a particular configuration if, and only if, she has a winning strategy from (q_0,x_0) in the VASS. The key property is that each player must be able to win whenever his adversary makes a counter value become negative. We can then simulate the VASS semantics.

In order to have this property, we construct a counter system (Q',T'), with vertices $Q' = Q \cup \{\text{test}_t \mid t = (q,v,r) \in T, v \notin \mathbb{N}^d\} \cup \{\bot, \text{check}, \text{check}_1, \ldots, \text{check}_d\}$. We partition Q' into $Q'_E = Q_E \cup \{\bot, \text{check}, \text{check}_1, \ldots, \text{check}_d\} \cup \{\text{test}_t \mid t = (q,v,r) \in T, q \in Q_A\}$ and Q'_A. The set of edges T' is obtained from T, first by splitting every edge $t = (q,v,r)$ such that $v \notin \mathbb{N}^d$ into two edges (q,v,test_t) and $(\text{test}_t, 0^d, r)$, and second by adding moves from every vertex test_t to check vertices of Q' as well as additional edges in the set of vertices that gathers check vertices and $\bot$, according to the list that we give and as depicted in Figures 2.6 and 2.7.

More precisely, T' is the union of the following sets of edges, where $(x)_i^d$ is the vector with x in the i^{th} component and 0 in the other components:

- $\{(q,v,r) \in T \mid v \in \mathbb{N}^d\}$;
- $\{(q,v,\text{test}_t), (\text{test}_t, 0^d, r) \mid t = (q,v,r) \in T, v \notin \mathbb{N}^d\}$;
- $\{(\text{test}_t, 0^d, \text{check}) \mid t = (q,v,r) \in T, v \notin \mathbb{N}^d, q \in Q_E\}$;
- $\{(\text{test}_t, (1)_i^d, \text{check}_i) \mid t = (q,v,r) \in T, q \in Q_A, 1 \leq i \leq d\}$;

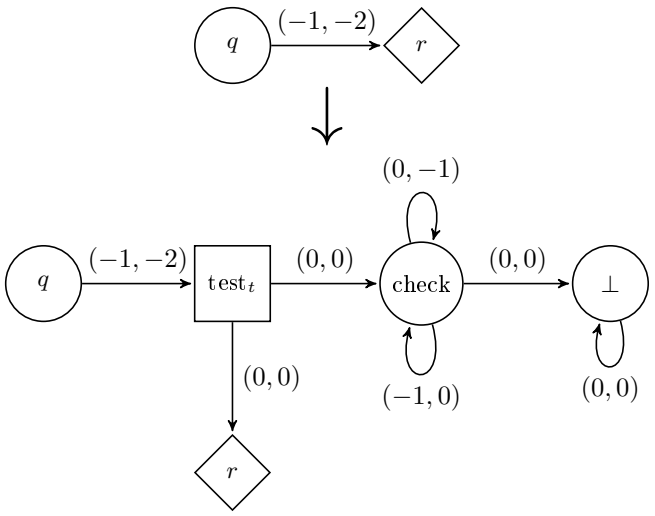

Figure 2.6 – Gadget to replace an edge $t = (q, (-1, -2), r)$ from a vertex of Eve in the reduction from $\text{VASS}_2((0,0))$ to $\text{CS}_2((0,0))$.

- $\{(\text{check}, (-1)_i^d, \text{check}) \mid 1 \leq i \leq d\}$;
- $\{(\text{check}_i, (-1)_j^d, \text{check}_i) \mid 1 \leq j \leq d, j \neq i\}$;
- $\{(\text{check}_i, (1)_j^d, \text{check}_i) \mid 1 \leq j \leq d\}$;
- $\{(q_f, -x_f, \bot)\} \cup \{(q, 0^d, \bot) \mid q \in \{\bot, \text{check}, \text{check}_1, \ldots, \text{check}_d\}\}$.

The objective of the counter reachability game is $(\bot, (0, \ldots, 0))$. Hence, in the vertex check, Eve has a winning strategy if, and only if, every counter is nonnegative, and in the vertex check_i, Eve has a winning strategy if, and only if, the i^{th} counter, which has been incremented when the play reached check_i, is nonpositive. Consequently, as soon as a player makes a counter become negative, his adversary has a winning strategy by going to a check vertex. If all counters remain positive, then Eve wins by using the move $(q_f, -x_f, \bot)$ once the play visits the objective of the game on (Q, T).

The reduction is polynomial: we have $|Q'| \leq d+2+|Q|+|T|$ and $|T'| \leq (d+2)|T|+2d(d+1)+2$. Moreover, if (Q, T) is unary, then (Q', T') is unary too, provided that the objective in the VASS is a vector that contains only values in $\{-1, 0, 1\}$, and otherwise we may split the edge that subtracts the objective to still get a reduction such that the system that we build is unary, even if the reduction is then no longer polynomial. □

Theorem 9: CS_2^1 *is undecidable.*

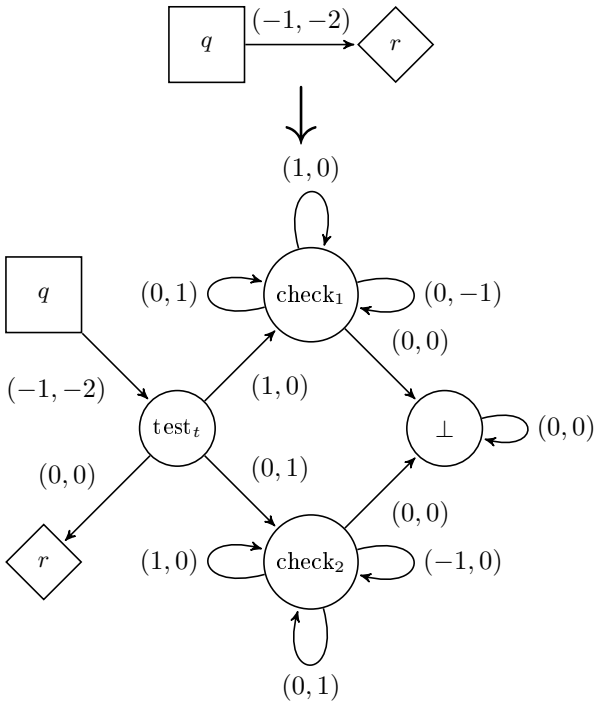

Figure 2.7 – Gadget to replace an edge $t = (q, (-1, -2), r)$ from a vertex of Adam in the reduction from $\textsc{Vass}_2((0,0))$ to $\textsc{Cs}_2((0,0))$.

Proof. We make two successive reductions: from the decision problem on VASS that is proved to be undecidable in Theorem 1 on page 21 to $\textsc{Vass}_2^1$ using Proposition 3 on page 21, then from $\textsc{Vass}_2^1$ to $\textsc{Cs}_2^1$ using Proposition 8. □

2.2.3 The two-player version in dimension one

We recall that Proposition 8 on page 29 implies that there is a polynomial-time reduction from $\textsc{Vass}_1^1(0)$ to $\textsc{Cs}_1^1(0)$, hence $\textsc{Cs}_1^1(0)$ is PSPACE-hard because $\textsc{Vass}_1^1(0)$ is PSPACE-complete according to [Ser06, BJK10].

The main idea of the construction in this section to prove PSPACE-membership of $\textsc{Cs}_1^1(0)$ by a reduction to $\textsc{Vass}_1^1(0)$ is to simulate, with nonnegative integers only, a counter value in $\mathbb{Z}$. For this purpose, we use two copies of the set of vertices and explain how to move from one copy to another.

Let us present the intuition of our reduction. We start from a counter system (Q, E) under the $\mathbb{Z}$ semantics and we build a counter system (Q', E') under the VASS semantics. One half of Q' represents locations of (Q, E) with nonnegative counter values in a corresponding run, the other half represents nonpositive counter values. Plays on (Q', E') correspond to plays on (Q, E). During a play on (Q', E'), any player can decide to move from one half to another one, provided that the counter value in the corresponding play on (Q, E) has the adequate sign. If a player makes such a decision unduly, then his adversary has a winning move in reaction. Winning moves are explained in the figures that depict the gadgets that the construction uses. With the feature of moving between the copies, it is possible to encode an integer into a pair (copy number, nonnegative integer).

Theorem 10: $\textsc{Cs}_1^1(0)$ *is* PSPACE*-complete.*

Proof. We reduce $\textsc{Cs}_1^1(0)$ to $\textsc{Vass}_1^1(0)$ in polynomial time. Consider a reachability game on a unary counter system (Q, T), where the objective is $(f, 0)$, with $f \in Q_E$. Recall that when the counter value in the objective is not 0, we can translate initial and objective value thanks to the $\mathbb{Z}$ semantics.

We define two copies $Q_+ = \{q_+ \mid q \in Q\}$ and $Q_- = \{q_- \mid q \in Q\}$ of Q, and the set $Q_T = \{q_t \mid \exists p, q \in Q, v \in \{\pm 1\}, t = (p, v, q) \in T\}$. We build the unary VASS (Q', T'), where $Q' = Q_+ \cup Q_- \cup Q_T \cup \{\text{no}, \bot\}$ is partitioned into $Q'_E = \{q_+, q_- \mid q \in Q_E\} \cup \{q_t \in Q_T \mid t \in Q_A \times \{\pm 1\} \times Q\} \cup \{\text{no}, \bot\}$ and Q'_A. The set of edges T' contains two copies of T, i.e., edges (q_+, v, r_+) and $(q_-, -v, r_-)$ for each edge $(q, v, r) \in T$. When $v = 0$ in the edge, we do not use any gadget and the play remains in the same copy of Q. The other edges of T' are used to move between Q_+ and Q_- via the new vertices of Q_T, as depicted in Figures 2.8 and 2.9.

More precisely, T' is the union of the following sets of edges:

- $\{(q_+, v, r_+), (q_-, -v, r_-) \mid (q, v, r) \in E\}$;

- $\{(q_-, 0, r_t), (q_t, 0, \bot), (q_t, 1, q_+) \mid t = (q, 1, r) \in T, q \in Q_E\}$;
- $\{(q_+, 0, r_t), (q_t, 0, \bot), (q_t, 1, q_-) \mid t = (q, -1, r) \in T, q \in Q_E\}$;
- $\{(q_-, 0, r_t), (q_t, -1, \text{no}), (q_t, 1, q_+) \mid t = (q, 1, r) \in T, q \in Q_A\}$;
- $\{(q_+, 0, r_t), (q_t, -1, \text{no}), (q_t, 1, q_-) \mid t = (q, -1, r) \in T, q \in Q_A\}$;
- $\{(\text{no}, -1, \text{no}), (\text{no}, 0, \bot), (f_+, 0, \bot), (f_-, 0, \bot), (\bot, 0, \bot)\}$.

The VASS (Q', T') is designed such that a play in it corresponds to a play in the counter system (Q, T). Let (q, x) be the initial configuration of the game on (Q, T). If $x \geq 0$, then the initial configuration of the game on (Q', T') is (q_+, x) ; if $x < 0$, then the initial configuration of the game on (Q', T') is $(q_-, -x)$. Hence, a configuration $(q, x) \in Q \times (-\mathbb{N})$ in (Q, T) is associated with the configuration $(q_-, -x) \in Q_- \times \mathbb{N}$ in (Q', T'). That is why the labels of the edges between vertices in Q_- are the opposite of the labels of the edges in Q.

The objective of the game on (Q', T') is $(\bot, 0)$. In fact, Eve loses whenever a play reaches $\bot$ with another counter value. Furthermore, if a player makes a move to a vertex q_t in Q_T and the counter value is not 0, then his adversary, who owns q_t, has a winning move. □

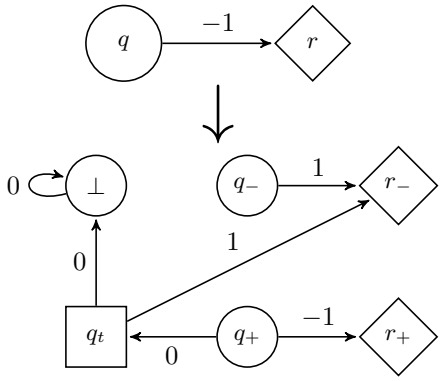

Figure 2.8 – Gadget to replace an edge $t = (q, -1, r)$ from a vertex of Eve in the reduction from $\text{CS}_1^1(0)$ to $\text{VASS}_1^1(0)$.

A consequence of Theorem 10 is that CS_1 is in EXPSPACE: It suffices to split every edge with another label than -1, 0 or 1. At the time when we wrote this section, we did not know yet whether EXPSPACE was an optimal upper bound, but a lower bound can be obtained from from the complexity of countdown games, presented in [JLS07], a model that we can express as counter reachability games.

Theorem 11 ([JLS07])**:** CS_1 *is* EXPTIME*-hard.*

According to Paul Hunter in [Hun14], the problem of determining the winner of a counter

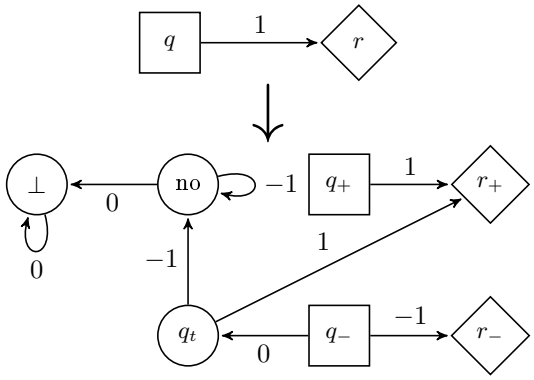

Figure 2.9 – Gadget to replace an edge $t = (q, 1, r)$ from a vertex of Adam in the reduction from $\text{CS}_1^1(0)$ to $\text{VASS}_1^1(0)$.

reachability game under any semantics in dimension one is EXPSPACE-hard. It appears that all complexities for the two-player game match with the corresponding complexities under the VASS semantics, unlike in the one-player case.

Note that the complexity result under the $\mathbb{Z}$ semantics holds whichever counter value is in the objective, as we explained at the beginning of the section. Since our reductions, although written with zero as objective, hold for any fixed counter value in the objective, we can extend our results to obtain PSPACE-completeness in dimension one under the other semantics. We must in fact take care that, on unary counter systems, when the counter value in the objective is $k \neq 0$, we need to create k fresh vertices and edges, which would have an effect on the complexity if k were in input.

Let us also extend our results to objectives that are not singletons but unions of $\{q_i\} \times X_i$, where all X_i are linear sets, with a similar, but more advanced, argument to the one in the proof of Proposition 4 on page 22. On non-unary systems, determining the winner of a reachability game under the $\mathbb{Z}$ semantics has the same complexity with and without singleton objectives, and on unary systems, the complexity does not differ either when it is already at least EXPTIME.

Proposition 12: *Let (Q, T) be a counter system under the $\mathbb{Z}$ semantics. Consider a reachability game on (Q, T), where the objective for Eve is a union of $\{q_i\} \times X_i$, where the $q_i \in Q$ and all $X_i \subseteq \mathbb{Z}^d$ are linear sets. We can build in polynomial time a counter system (Q', T') under the $\mathbb{Z}$ semantics in which the objective of Eve is a singleton and such that Eve has a winning strategy from a given initial configuration (q_0, x_0) in (Q, T) if, and only if, she has a winning strategy from the same initial configuration in (Q', T').*

Proof. Let (Q, T) be a counter system under the $\mathbb{Z}$ semantics, let $q_1, \ldots, q_n$ be elements of Q and for $1 \leq i \leq n$ let $X_i = x_i + x_{i,1}\mathbb{N} + \cdots + x_{i,k_i}\mathbb{N}$ be linear sets. For the case of semilinear sets, i.e., finite unions of linear sets, we just consider that a same vertex can appear many times in the list of q_i.

We suppose without loss of generality that all q_i belong to Eve, like in the proof of Proposition 3 on page 21. We build a counter system (Q', T') under the $\mathbb{Z}$ semantics, where $Q' = Q \cup \{q'_1, \ldots, q'_n, \bot\}$, $Q'_E = Q' \setminus Q_A$, and T' is the union of the following sets of edges:

- T;
- $\{(q_i, 0^d, q'_i) \mid 1 \leq i \leq n\}$;
- $\{(q'_i, -x_{i,j}, q'_i) \mid 1 \leq i \leq n, 1 \leq j \leq k_i\}$;
- $\{(q'_i, -x_i, \bot) \mid 1 \leq i \leq n\} \cup \{(\bot, 0^d, \bot)\}$;

The objective in (Q', T') is the singleton $(\bot, 0^d)$.

Intuitively, a play in (Q', T') simulates the same play in (Q, T) up to a point when Eve claims that she would win in (Q, T). Indeed, whenever the play is in a q_i and the counter vector is in X_i, Eve may force to reach q'_i, where she subtracts vectors that are directions of X_i until the counter vector is the base vector of X_i. By subtracting this base vector and going to $\bot$, Eve wins.

Conversely, by a careful look at T', we remark that Eve wins a play in (Q', T') if, and only if, at some point in the play, just before entering a q'_i, the configuration was (q_i, x) with $x \in X_i$, and before this point plays in (Q, T) and in (Q', T') were the same, insofar as the same configurations are visited.

Finally, note that the reduction that we present is polynomial, which is useful when we need the result of this proposition for dimension one. Indeed, the number of additional vertices is the number of linear sets in the objective plus one, and the number of additional edges is linear in the number of base vectors in all linear sets. However, with this reduction, we do not build in general unary systems, and building unary systems would cause an exponential blowup insofar as the new edges would have to be split. It is only if all vectors in all X_i have only components in $\{-1, 0, 1\}$ that the systems that we build according to the reduction are unary. □

2.3 Reachability games on counter systems under the Non-blocking VASS semantics in dimension one

When we consider possible behaviours around zero in counter systems, we understand that between the $\mathbb{Z}$ semantics, which always allows to take every edges, and the VASS semantics, under which an edge is disabled if it would make a counter become negative, there may be an intermediary choice: if an edge would make at least one counter become negative, it is indeed crossed, but the counters that should have become negative are put to zero instead.

This leads to the definition of a new semantics, that we call non-blocking VASS semantics in [Rei13].

A counter system under the non-blocking VASS semantics has a weaker version of the regularity property that counter systems under the $\mathbb{Z}$ semantics have: Consider a run in the underlying graph, the effect of the run on the counters in the system will not necessarily be the same for two different initial counter vectors. In other words, the difference between final and initial counter vectors may not be preserved when the initial counter vector is changed. Nevertheless it is guaranteed that if a counter is smaller in the first initial configuration than in the second initial configuration, then the same counter will be smaller or equal in the first final configuration than in the second final configuration.

In this section, the counter systems are all in dimension one.

When counter values are far away from 0, there is no difference between the two semantics, and we first show how to simulate the VASS semantics with the non-blocking VASS semantics and vice-versa, when the counter value in the objective of the game under the non-blocking VASS semantics is a constant. According to the mutual reductions that we obtain, determining the winner of a counter reachability game under the non-blocking VASS semantics with a positive objective given in unary is PSPACE-complete.

In the specific case where the objective is 0, and because of the above property, the winning set for Eve is downward closed, hence we may compute for every vertex the maximal starting value from which she has a winning strategy, which leads to a polynomial-time algorithm for determining the winner if the system is unary. With non-unary systems, the problem is in NP.

Let us first deal with the reachability problem under the non-blocking VASS semantics. Recall that this problem is NP-complete in dimension one for counter systems under the $\mathbb{Z}$ semantics and under the VASS semantics. Here, knowing the counter value before taking an edge is even more important than in VASS, because the effect of an edge changes when the value is near zero. Hence we do not look for a direct algorithm for solving the reachability problem, but rather for a reduction from the same problem under the VASS semantics. In fact, what matters is whether an edge that decreases the counter, say by k, is taken, whereas the counter value is lower than k. Accordingly, consider a run ρ from a configuration (q, x) to a configuration (q_f, x_f) in a counter system (Q, T) under the non-blocking VASS semantics. For all edges $t = (r, v, s) \in T$ such that $v < 0$, we can assume without loss of generality that in ρ the edge t is taken at most once while the counter value is $\leq v$, because a witness of reachability need not visit twice the same configuration, and taking t from any configuration (r, y) with $y \leq v$ leads to $(s, 0)$.

Theorem 13: *The reachability problem for one-dimensional counter systems under the non-blocking* VASS *semantics is in* NP.

Proof. **(NP-membership)** Let us reduce the reachability problem for one-dimensional counter systems under the non-blocking VASS semantics to the same problem under the VASS semantics.

Let (Q, T) be a one-dimensional counter system under the non-blocking VASS semantics. Let $(q, x), (q_f, x_f) \in Q \times \mathbb{N}$ be two configurations.

As stated before the proposition, if there is a run in (Q,T) from (q,x) to (q_f,x_f), then each edge that decreases the counter can be taken at most once from a configuration such that the counter value then becomes zero. Let $Q_0 \subseteq Q$ be the set of targets of edges that decrease the counter.

Apart from that, the behaviour of the system is like under the VASS semantics. Hence, for each vertex $r \in Q_0$, we would like to know whether (q_f,x_f) is reachable from $(r,0)$ in (Q,T), but under the VASS semantics, which amounts to solving the reachability problem.

We then make the connections between parts of a run candidate from (q,x) to (q_f,x_f) in (Q,T) under the non-blocking VASS semantics, linking the parts in $(r,0)$ for $r \in Q_0$. To do this, we guess an ordered sequence $r_1,\dots,r_n$ of elements of Q_0, each of them appearing at most once, and we guess the subruns from (q,x) to $(r_1,0)$, from $(r_i,0)$ to $(r_{i+1},0)$ for $1 \leq i < n$ and from $(r_n,0)$ to (q_f,x_f).

For the last subrun, we already explained how to check reachability. For the previous ones, we need to modify the counter system in order to simulate the last edge taken in the run, because this edge would be disabled under the VASS semantics and we cannot replace it by multiple edges because the reduction would no longer be polynomial.

Hence, instead of determining whether there is a run from $(r_i,0)$ to $(r_{i+1},0)$ in (Q,T) under the VASS semantics (analogous for the first subrun), we decide whether there is a run from $(r_i,0)$ to $(\bot,0)$ in $(Q \cup Q' \cup \{\top\},T')$ under the VASS semantics, where $Q' = \{\top_q \mid (q,v,r_{i+1}) \in T \cap Q \times (-\mathbb{N}) \times Q\}$ is a copy of the set of sources from edges that decrease the counter and have r_{i+1} as targets, $T' = T \cup \{(q,0,\top_q),(\top_q,1,\top_q) \mid q \in Q'\} \cup \{(\top_q,v,\top) \mid (q,v,r_{i+1}) \in T\} \cup \{(\top,0,\top)\}$ contains the edges in T and for each vertex in $\top_q \in Q'$ a gadget that simulates the edge from q to r_{i+1} and decreases the counter to zero. In this gadget, the counter is incremented as many times as needed such that when $\top$ is reached the value is 0. Of course, the gadget should be entered when the counter value is low enough.

Finally, we check whether the guessed subruns exist in the modified counter systems under the VASS semantics, which leads to the announced complexity.

(NP-hardness)

The NP lower bound uses a straightforward reduction from the SUBSET-SUM problem like the case of the other two semantics. In the reduction for this semantics, it is necessary that the reachability problem is whether there is a path from a configuration $(q_0,0)$ to a configuration (q_f,s), where q_0 is the initial vertex, q_f is the final vertex and s is the sum in the instance of SUBSET-SUM. Indeed, if 0 were the counter value in the objective, then this value could be reached while subtracting more than s, under the non-blocking VASS semantics. As we evoked before, when the value is 0 in the objective, the complexity of determining the winner of a reachability game is lower than in the general case, which explains why the same reduction cannot hold. □

2.3.1 Reachability games on non-blocking VASS with value 1 in the objective

In this subsection, we give a reduction from the problem of determining the winner on a non-blocking VASS to the problem of determining the winner on a VASS, and also the reverse reduction. The idea of the first reduction is the following. For every edge labelled by -1 in a unary non-blocking VASS, we give two choices for Adam in the VASS: decrement the counter or leave it unchanged, depending on whether it is positive or zero. The winning condition is designed so that Eve has a checking move that makes her win whenever Adam chooses the wrong move, e.g., he leaves the counter unchanged whereas he should decrement it. Moreover, Adam wins if Eve abuses her checking move.

Theorem 14: $\text{NBVASS}_1^1(1)$ *is* PSPACE-*complete.*

Proof. **(PSPACE-membership)** We reduce $\text{NBVASS}_1^1(1)$ to $\text{VASS}_1^1(0)$ in polynomial time. Let (Q,T) be a unary non-blocking VASS (Q,T). Consider a reachability game on (Q,T), where the objective is to reach $(q_f,1)$ for a fixed location $q_f \in Q_1$. We define Q_T as the set $\{q_t, q_t^{>0}, q_t^{=0} \mid t \in T \cap (Q \times \{-1\} \times Q)\}$, and we build the unary VASS (Q',T'), where $Q' = Q \cup Q_T \cup \{\text{no}, \bot\}$ is partitioned into $Q'_E = Q_E \cup \{q_t^{>0}, q_t^{=0} \mid t \in E\} \cup \{\text{no}, \bot\}$ and Q'_A. The set of edges is

$$\begin{aligned} T' = &\{(q,v,r) \mid (q,v,r) \in T, v \in \{0,1\}\} \\ &\cup \{(q,0,q_t), (q_t,0,q_t^{>0}), (q_t,0,q_t^{=0}), (q_t^{=0},0,r), (q_t^{>0},-1,r), \\ &\quad (q_t^{>0},0,\bot), (q_t^{=0},-1,\bot) \mid t = (q,-1,r) \in T\} \\ &\cup \{(\text{no},-1,\text{no}), (\text{no},0,\bot), (q_f,-1,\bot), (\bot,0,\bot)\}. \end{aligned}$$

Intuitively, every time a play visits an edge with a decrement in (Q',T'), Adam has to guess whether the counter value is zero or positive, and move accordingly to an intermediate vertex, where Eve can move to the actual target of the edge in (Q,T) or to a checking gadget where the play ends.

The objective of the game on (Q',T') is $(\bot,0)$. As we can see in Figure 2.10, Eve has a winning strategy in every vertex $q_t^{=0}$ when the counter value is positive, and in every vertex $q_t^{>0}$ when the counter value is zero.

In the construction for the reverse reduction, when a player chooses any edge with a negative label and the counter value is less than the value that should be subtracted, then the adversary of this player has a winning move. While taking the aforementioned edge is allowed in a non-blocking VASS, it would be forbidden in a VASS.

(PSPACE-hardness) We show a polynomial-time reduction from $\text{VASS}_1(0)$ to $\text{NBVASS}_1(1)$. Let (Q,T) be a VASS. Consider a reachability game on (Q,T), where the objective is $(q_f,0)$, with $q_f \in Q_E$. We define Q_T as the set $\{q_t \mid t \in T \cap (Q \times (\mathbb{Z} \setminus \mathbb{N}) \times Q)\}$, and we build the non-blocking VASS (Q',T'), where $Q' = Q \cup Q_T \cup \{\text{no}_\text{E}, \text{no}_\text{A}, \bot\}$, $Q'_E = Q_E \cup \{q_t \in Q_T \mid t \in Q_2 \times \mathbb{Z} \times Q\} \cup \{\text{no}_\text{E}, \text{no}_\text{A}, \bot\}$, $Q'_A = Q' \setminus Q'_E$, and T' is obtained from T by splitting every edge

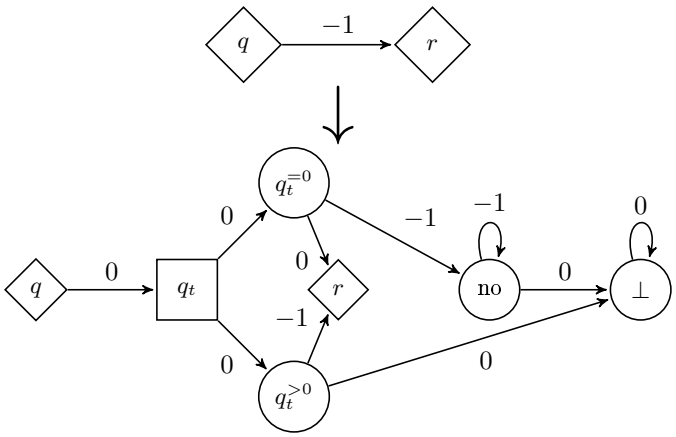

Figure 2.10 – Gadget to replace an edge $t = (q, -1, r)$ in the reduction from $\text{NBVASS}_1^1(1)$ to $\text{VASS}_1^1(0)$.

(q, v, r) such that $v \in -\mathbb{N}$ into two edges $(q, 0, q_t)$ and (q_t, v, r) and by adding an edge from every vertex q_t to the "no"-vertex that corresponds to the owner of q, as well as additional edges between no_A, no_E and $\bot$, as depicted in the Figures 2.11 and 2.12.

More precisely, T' is the union of the sets of edges:

- $\{(q, v, r) \mid (q, v, r) \in T, v \in \mathbb{N}\}$;
- $\{(q, 0, q_t), (q_t, v, r) \mid t = (q, v, r) \in T, v < 0\}$;
- $\{(q_t, v + 1, \text{no}_\text{E}) \mid t = (q, v, r) \in T, v < 0, q \in Q_E\}$;
- $\{(q_t, v + 1, \text{no}_\text{A}) \mid t = (q, v, r) \in T, v < 0, q \in Q_A\}$;
- extra edges $\{(\text{no}_\text{E}, -1, \text{no}_\text{E}), (\text{no}_\text{E}, 0, \bot), (\text{no}_\text{A}, 1, \bot), (q_f, 1, \bot), (\bot, 0, \bot)\}$.

The non-blocking VASS (Q', T') is designed such that a play in it corresponds to a play in the VASS (Q, T). Let us consider a vertex $q_t \in Q_T$, for an edge (q, v, r) in T. Note that $v < 0$ and that the owner of q_t is not the owner of q. In the play on the VASS, the edge (q, v, r) can only be taken if the counter value is at least $-v$. If a player goes to q_t, i.e., simulates the choice of the edge (q, v, r), his adversary wins whenever the counter value is less than $-v$, after going to a "no"-vertex, as we can see in the Figures 2.11 and 2.12.

Indeed, suppose that Eve tries to take an edge t that decreases the counter by 5 whereas the counter value is less. Adam goes to no_E and decreases the counter by 4, so the counter becomes 0 because of the non-blocking VASS semantics. In no_E, Eve cannot decrement the counter and bring its value to 1, hence she cannot reach her objective, which is $(\bot, 1)$. Now, suppose that Adam tries to take an edge t that decreases the counter by 5 whereas the counter value is less. Eve goes to no_A and decreases the counter by 4, so the counter also becomes 0. Then Eve wins at the next move

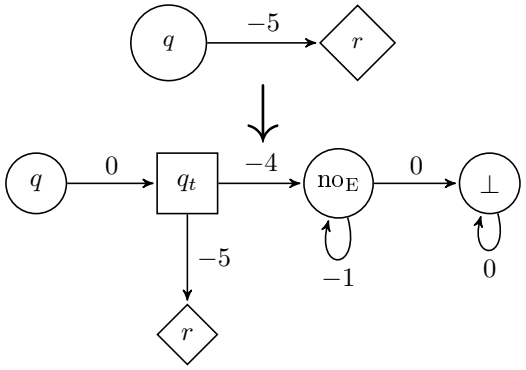

Figure 2.11 – Gadget to replace an edge $t = (q, -5, r)$ from a vertex of Eve in the reduction from $\text{VASS}_1(0)$ to $\text{NBVASS}_1(1)$.

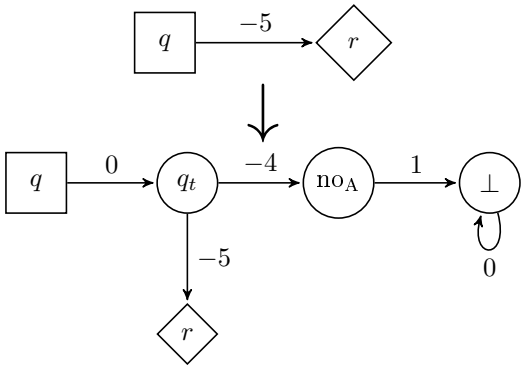

Figure 2.12 – Gadget to replace an edge $t = (q, -5, r)$ from a vertex of Adam in the reduction from $\text{VASS}_1(0)$ to $\text{NBVASS}_1(1)$.

because the configuration is $(\bot, 1)$. □

Note that according to this reduction, if (Q, T) is a unary VASS, then (Q', T') is a unary non-blocking VASS.

2.3.2 The case of zero-reachability games on non-blocking VASS

For non-blocking VASS, we prove that the set of winning configurations is downward closed when the reachability objective is $(q_f, 0)$ for a given q_f. Hence, to decide whether Eve has a winning strategy, we compute for all vertices the maximal initial value for which the pair (vertex, value) is winning and we look at the initial configuration.

Lemma 15: *Let (Q, T) be a non-blocking* VASS*. Consider a reachability game on (Q, T), where the objective is $(q_f, 0)$, where $q_f \in Q$. If the initial configuration (q_0, x) is winning, then every configuration (q_0, x') for $x' < x$ is winning.*

Proof. Let (q_0, x) be a winning configuration, and let s be a winning strategy for Eve from (q_0, x). We want to prove that s is also a winning strategy from (q_0, x') for $x' < x$. Consider any strategy s' for Adam. The outcome of the strategies s and s' from (q_0, x) is a play π that Eve wins, i.e., the play π eventually visits $(q_f, 0)$, because s is a winning strategy. Also, the outcome of the same strategies from (q_0, x') is a play π' that visits the same locations as π, and no edge is disabled because of the semantics of a non-blocking VASS. Moreover, the counter value in π' is after each move less than or equal to the counter value in the corresponding move of π. In particular, π' eventually visits q_f with counter value 0, hence Eve wins. □

Algorithm 1 determines whether Eve has a winning strategy in a reachability game on a non-blocking VASS when the counter value in the objective is 0, by computing for every vertex q the greatest counter value x such that Eve has a winning strategy from (q, x). Its time complexity is exponential in the initial counter value. Because we want a polynomial-time complexity, we call it only with 0 as initial counter value in the proof of Theorem 16.

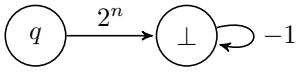

Figure 2.13 – Counter system where Algorithm 1 uses 2^n iterations for an input of size n.

Theorem 16: $\text{NBVASS}_1^1(0)$ *is* P*-complete.*

Proof. According to Lemma 15, we just need to compute for every vertex $q \in Q$ the maximal value x_m such that (q, x_m) is winning. We even do more: First, we compute the set Q_Z of vertices from which Eve has a winning strategy with initial counter value 0. For this purpose, we use Algorithm 1, and here the time complexity is polynomial. Second, we build the VASS (Q', T'),

Algorithm 1: Solves $\text{NBVASS}_1^1(0)$.

Input: A non-blocking VASS (Q, T), a vertex q_f, and a configuration (q_0, x_0)
Result: Does Eve have a winning strategy to reach $(q_f, 0)$ from (q_0, x_0)?
begin
  Create a table M_q with $q \in Q$ as indices, initialized to $-\infty$
  $M_{q_f} \leftarrow 0$
  repeat
    foreach $q \in Q_E$ **do**
      $new\_M_q \leftarrow -\infty$
      if $q = q_f$ **then** $new\_M_q \leftarrow 0$
      foreach $t = (q, v, r) \in E$ **do**
        if $M_r - v \geq 0$ **then** $new\_M_q \leftarrow \max(new\_M_q, M_r - v)$
    $M_q \leftarrow new\_M_q$
    foreach $q \in Q_A$ **do**
      $new\_M_q \leftarrow \infty$
      foreach $t = (q, v, r) \in T$ **do**
        if $M_r - v \leq 0$ **then** $new\_M_q \leftarrow -\infty$
        else $new\_M_q \leftarrow \min(new\_M_q, M_r - v)$
      if $q = q_f$ **then** $new\_M_q \leftarrow \max(0, new\_M_q)$
    $M_q \leftarrow new\_M_q$
  until *a fixpoint is reached or* $M_{q_0} \geq x_0$
  if $M_{q_0} \geq x_0$ **then return** *true*
  else return *false*

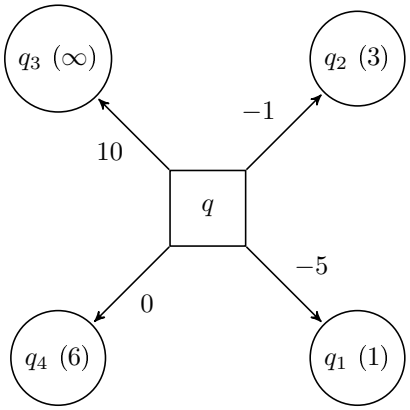

Figure 2.14 – Vertex of Adam in a game on a non-blocking VASS (maximal winning value for Eve in parenthesis).

Value in q	≤ 3	4 or 5	≥ 6
If edge to q_1	No		Yes
If edge to q_2	No	Yes	
If edge to q_3	No		
If edge to q_4	No		Yes

Figure 2.15 – Does Adam have a winning strategy in a zero-reachability game on a non-blocking VASS?

where $Q' = Q_Z \cup \{\bot\}$ and T' is the union of $T \cap (Q_Z \times \mathbb{Z} \times Q_Z)$ and of $\{(q, 1, \bot) \mid (q, v, r) \in T, q \in Q_Z, r \notin Q_Z\} \cup \{(\bot, 0, \bot)\}$. In (Q', T'), the value 0 can only be reached in a vertex that belongs to Q_Z. Consider the reachability game on (Q', T'), where the objective is $Q \times \{0\}$, like defined in [BJK10]; determining the winner in this game is in P. Moreover, Eve has a winning strategy if, and only if, she has a strategy in Q to reach $(q, 0)$ for any $q \in Q_Z$, hence to reach $(q_f, 0)$. Indeed, if a play visits a vertex outside of Q_Z, then Adam has a winning strategy. We conclude that determining the winner of the reachability game is in P too. □

Let us extend this result to counter systems which are not unary. Algorithm 1 should be modified to detect negative cycles and it should bound in a more clever way the number of iterations. As we use a binary encoding of the input, the number of iterations would be exponential for a simple counter system like in Figure 2.13.

Theorem 17: $\text{NBVASS}_1(0)$ *is in* NP.

Proof. We suppose that Eve has a strategy to reach q_f from each vertex, in particular Adam cannot force any cycle. If it is not the case, we may as well replace some vertices by sinks.

We use another algorithm, Algorithm 2, that also computes from each vertex the greatest initial value for which Eve has a winning strategy, according to a similar principle to the one of shortest path algorithms. The greatest value that we compute increases at each iteration until either a fixpoint is reached or the initial configuration is proved to be winning. However, unlike for the unary case, we use the algorithm alone, in the one-player case (as explained in the following) and not with 0 as input, hence we need to detect negative cycles. To do this, we keep for each vertex

Algorithm 2: Solves $\text{NBVASS}_1(0)$ in the one-player case.

Input: A non-blocking VASS (Q, T), a vertex q_f, and a configuration (q_0, x_0)
Result: Does there exist a run from (q_0, x_0) to $(q_f, 0)$?
begin
  Create a table M_q with $q \in Q$ as indices, initialized to $-\infty$
  Create a table P_q with $q \in Q$ as indices, initialized to $\{\varepsilon\}$
  $M_{q_f} \leftarrow 0$
  $P_{q_f} \leftarrow \{q_f\}$
  repeat
    foreach $q \in Q$ **do**
      foreach $t = (q, v, r) \in T$ **do**
        if $M_q < M_r - v$ *AND* $M_r - v > 0$ *// convention: test is false when* ∞ *appears on left hand or* $-\infty$ *on right hand*
        then
          if q *is in a sequence of* P_r *// the cycle can only be negative*
          then
            $M_q \leftarrow \infty$
            $P_q \leftarrow \{\perp\}$ // $\perp$ is not a location, but indicates that there is no need to keep the information
          else if $P_r = \{\perp\}$ *// hence* M_r *is* ∞
          then
            $M_q \leftarrow \infty$
            $P_q \leftarrow \{\perp\}$
          else
            $M_q \leftarrow M_r - v$
            $P_q \leftarrow \emptyset$
            foreach $seq_1 \in P_r$ *// append the vertex to the sequence*
            do
              $\text{seq} \leftarrow q\ \text{seq}_1$
              $P_q \leftarrow P_q \cup \{\text{seq}\}$
        if $M_q = M_r - v$
        then
          foreach $seq_1 \in P_r$ **do**
            if q *is not in* seq_1 *// else the cycle is zero and we ignore it*
            then
              $\text{seq} \leftarrow q\ \text{seq}_1$
              if *seq contains an element in no sequence of* P_q **then**
                $P_q \leftarrow P_q \cup \{\text{seq}\}$
  until *a fixpoint is reached or* $M_{q_0} \geq x_0$
  if $M_{q_0} \geq x_0$ **then return** *true*
  else return *false*

information not only about the greatest winning value but also about a list of paths to the vertex q_f for the optimal value. The paths that we store for a given vertex bear the following constraints:

- In every path, no vertex appears twice, or else we have necessarily a nonpositive cycle. We compute the value of the cycle: if it is zero, then we remove it; else we know that the vertex is winning for any initial value and we set this to ∞, after that we do not need to remember the paths anymore for this vertex.

- Every path contains a vertex that appears in no other path, else it would not be useful to store it.

Because of this, the number of paths that we keep for each vertex is at most the number of vertices, and the total size of the information is cubic.

Let us prove the correctness of Algorithm 2. The algorithm ends when no entry of the table M_q changes upon an iteration on all vertices, or when a positive answer is obtained. For each iteration, for every vertex q, all edges with q as source are considered, which causes possible updates on the value M_q, so the maximal value x for which it is established that there exists a run from (q, x) to $(q_f, 0)$. Recall that we only deal with the one-player case.

The first possibility for an edge (q, v, q') is that the target vertex improves the maximal winning value, in other words $M_{q'}$ is positive and $M_{q'} > M_q + v$, which is equivalent to the conjunction $M_q < M_{q'} - v$ and $M_{q'} - v > 0$. In this case, M_q should be adapted and the memory of optimal paths should be updated by adding q at the beginning of all optimal paths from q'. Also, this may lead to the detection of a negative cycle, if q was already in an optimal path. With the possibility of crossing a negative cycle, every nonnegative integer becomes winning, which justifies the value ∞. For the same reason, if the value of q' is ∞, then the value of q can be updated to ∞. With such a value, it is no longer relevant to remember optimal paths, negative cycles need not be detected.

The second possibility is that the target vertex gives another way to have the same maximal winning value, in other words $M_{q'} = M_q + v$, which is equivalent to $M_q = M_{q'} - v$. In this case, for the purpose of detecting negative cycles, the set of optimal paths should take into account the new possible paths through q', which are relevant as soon as they have a vertex that belongs to no other optimal path.

For the third possibility, it is useless to consider the edge, at least as long as the value $M_{q'}$ is not modified.

Finally, upon reaching a fixpoint, the table M_q stores the maximal winning value for all vertices, so it suffices to make the test for q_0.

Note that the cycles that we detect should be forced by Eve, and so should be the paths of which we keep track. But Adam can also make choices at some points. We could handle this by considering boolean formulas over paths and consider satisfiability issues to decide whether a negative cycle is controllable, but it seems easier to use the following property of a game on non-

blocking VASS with objective 0: Adam has optimal memoryless strategies. Indeed, assume that he computes himself the greatest winning value for each vertex, and consider one of his vertices from which there are two edges, one with label k to a vertex with greatest winning value v, and one with label k' to a vertex with greatest winning value v'. It is always better for Adam to choose the first edge if $v - k < v' - k'$ and the second one if $v' - k' < v - k$, as depicted in Figure 2.14. In other words, Adam chooses the edge for which the difference is the smallest. If several differences are the smallest, he can always choose the same. Hence, we can restrict to memoryless strategies for Adam, and guess his strategy, at the cost of nondeterminism, thus obtaining a one-player game where the problem of controlling the loops disappears. □

For our new model of non-blocking VASS, we leave the complexity of the reachability problem, especially in arbitrary dimension, as an open problem. In the two-player case, the complexities match with the corresponding complexities under the other two semantics in the general case, with the notable exception of the problem of determining the winner when the counter value in the objective is zero.

2.4 Reachability games on reset counter systems

In this section, we consider one-dimensional counter reachability games on counter systems where some edges, labelled by $\mathbf{z}$, reset the counter to zero. We show how to decide the winner in such games with polynomially many calls to an algorithm that decides the winner in counter reachability games.

A *reset counter system* in dimension one is a directed graph (Q, T), where Q is a finite set of vertices and $T \subseteq Q \times (\mathbb{Z} \cup \{\mathbf{z}\}) \times Q$. Configurations and runs are defined like for counter systems, where runs may contain a subsequence $(q_i, x_i)(q_{i+1}, 0)$, for any $x_i \in \mathbb{Z}$, if $(q_i, \mathbf{z}, q_{i+1}) \in T$.

We can assume without loss of generality that reset counter systems have the property, which we call *isolation property*, that edges with a reset are the only outgoing edges of their source. Indeed, we just need to create new intermediary vertices that are reachable through edges with label 0, and from which either a reset edge or the non-reset edges start, respectively.

We call ResetCs$_1$ the problem of determining the winner of a reachability game on a reset counter system in dimension one.

Proposition 18: ResetCs$_1$ *reduces in polynomial time to* Cs$_1$.

Proof. Let us consider a reset counter system (Q, T) with the isolation property, and a reachability game played on (Q, T) with objective (q_f, x_f) and initial configuration (q_0, x_0). We denote by T_Z the set of reset edges in T and Q_Z the set of their sources. The idea of the reduction is the following: Suppose that Eve has a winning strategy, and consider a run from (q_0, x_0) to (q_f, x_f). This run goes through a finite number of reset edges, possibly zero. Also, we may find a run from

(q_0, x_0) to (q_f, x_f) such that each reset edge is visited at most once. Let us suppose that Eve tries to use as few reset edges as possible. We consider the counter reachability game played on $(Q', T') := (Q \cup \{\bot\}, T \setminus T_Z \cup T_\bot)$, where $\bot$ is a sink that makes Eve lose, $T_\bot = \{(q, 0, \bot) \mid q \in Q_Z\}$, and the objective is (q_f, x_f). Note that there is no reset in the game that we just defined: reset edges of (Q, T) have become losing edges for Eve. Because Eve has a winning strategy in (Q, T), she has a winning strategy in the game on (Q', T') from at least one of the configurations $(q, 0)$ for $q \in Q_Z$ or from (q_0, x_0). Indeed, Adam cannot force to take a reset edge infinitely often without that the objective of the game on (Q, T) is visited, because it would contradict the fact that Eve wins on (Q, T). Hence, starting either from the initial configuration or from a configuration just after a reset edge, Eve has a strategy such that the objective is reached while no further reset edge is taken.

This is the main principle that we use: We decide from which configurations in the set $C_Z := \{(q_0, x_0)\} \cup \{(q, 0) \mid q \text{ target of a reset edge}\}$ Eve has a winning strategy in a counter reachability game $(Q \cup \{\bot\}, T')$, where T' is T with every reset edge redirected to a sink $\bot$ where Eve loses. If the initial configuration is winning in $(Q \cup \{\bot\}, T')$, then we know that Eve has a winning strategy in (Q, T). Else, we redirect to a winning gadget all reset edges that have a target in the set of vertices q such that $(q, 0)$ is winning in $(Q \cup \{\bot\}, T')$. The gadget is a new vertex $\top$ that belongs to Eve and in which she has an incrementing loop, a decrementing loop and an edge labelled by 0 to q_f. We obtain yet another counter reachability game. We compute the set of the remaining configurations in C_Z that are now winning in this new game, and we repeat the two steps: redirecting edges and computing the winning configurations in the new game, until either the initial configuration becomes winning, hence Eve has a winning strategy, or a fixpoint is reached, i.e., no configuration becomes winning, hence Adam has a winning strategy. □

Let us illustrate this reduction with an example. Figure 2.16 depicts a reset counter system, and Figure 2.17 represents an equivalent one with the isolation property. Consider a reachability game on this system, where the objective is $(q_4, 3)$. The first step consists of redirecting the reset edges to a losing sink, as in Figure 2.18. In the counter system that we obtain, we call an algorithm that solves counter reachability games. The configuration $(q_1, 0)$ is winning for Eve, but the configuration $(q_0, 0)$ is not. The initial configuration of the reachability game on the initial reset counter system, that we do not need to specify, may actually be winning too. We get a new counter system for the second step in Figure 2.19 by redirecting the reset edge from vertex q_4 to a winning sink. At this point, we easily deduce that every configuration where the vertex is neither $q_{3'}$ nor $\bot$ is winning, and with a third iteration the losing sink becomes unreachable and Eve wins from any initial configuration. The procedure is even constructive because we can compute a winning strategy from winning strategies in reachability games without resets, by keeping in mind through which reset edges Eve needs to go.

Corollary 19: ResetCs$_1$ *is in* EXPSPACE *and* EXPTIME-*hard.*

According to the new lower bound from [Hun14], we even get a precise complexity.

Theorem 20: Cs$_1^{\mathbf{z}}$ *is* EXPSPACE-*complete.*

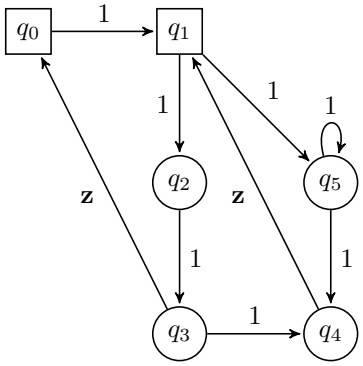

Figure 2.16 – Reset counter system.

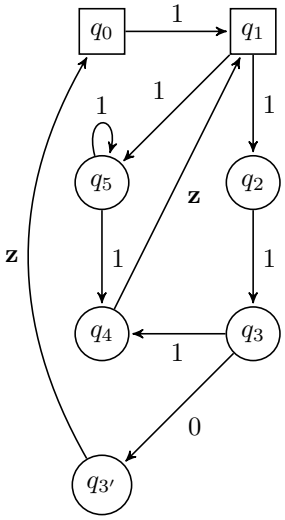

Figure 2.17 – Equivalent reset counter system with isolation property.

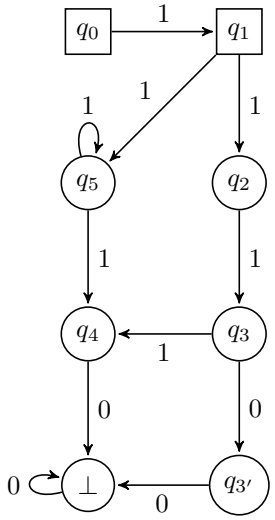

Figure 2.18 – First step.

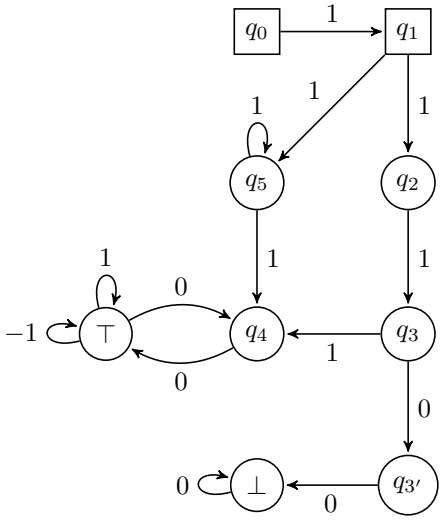

Figure 2.19 – Second step.

Note that, in fact, the semantics plays no role in the proof of Proposition 18. The reduction also works for the VASS semantics and for the non-blocking VASS semantics.

2.5 Affine updates

Until now, the counter systems that we consider have updates that consist of additions, and resets in one section. The common point between additions and resets is that they are affine functions, respectively $x \mapsto 1 \cdot x + v$ and $x \mapsto 0 \cdot x + 0$. In this section, we extend the updates of edges in counter systems to any affine function. A more general version of this model, with polynomial updates, has been studied in [FGH13], where several complexity results are given in dimension one.

To represent the label of an edge in this model, we write $ax + b$ in a component, where x is a variable and a, b are integers, if the edge applies the function $x \mapsto ax + b$ to the corresponding counter. In other words, in a d-dimensional counter system with affine updates, when an edge $(q, (a_1x + b_1, \ldots, a_dx + b_d), q')$ is taken from a configuration $(q, (x_1, \ldots, x_d))$, the next configuration is $(q', (a_1x_1 + b_1, \ldots, a_dx_d + b_d))$. Let us insist on the fact that, for example in dimension one, an edge with label k *sets* the counter value to k, and an edge that *adds* k has label $x + k$.

In dimension two, the reachability problem is already undecidable for affine updates with coefficients that are powers of 2. We can even restrict to coefficients 2 and 1 precisely, but the construction becomes heavier.

Proposition 21: *The reachability problem for two-dimensional counter systems with affine updates is undecidable.*

Before proving the proposition, let us introduce the *Post Correspondence Problem* (PCP), presented in [Pos46]. Given an alphabet Σ, and a set $\{(u_1, v_1), \ldots, (u_n, v_n)\}$ of pairs of words on Σ, decide whether there is a finite nonempty sequence $i_1, i_2, \ldots, i_k$ of elements of $\{1, \ldots, n\}$ such that the two words $u_{i_1}u_{i_2} \ldots u_{i_k}$ and $v_{i_1}v_{i_2} \ldots v_{i_k}$ are the same.

The intuition is that the pairs of words on Σ are dominos with one word in the up side and one word in the down side. The Post correspondence problem asks whether there is a sequence of dominos that give the same word in the up side as in the down side when we put them one next to the other.

Theorem 22 ([Pos46])**:** *PCP is undecidable when Σ contains at least two letters.*

Proof of Proposition 21. We reduce the Post Correspondence Problem to the reachability problem for two-dimensional counter systems with affine updates.

Let $\Sigma = \{0, 1\}$, let $D = \{(u_1, v_1), \ldots, (u_n, v_n)\}$ be a set of pairs of words on Σ. For $1 \leq i \leq n$, define h_i as the size of u_i and k_i as the size of v_i, define also $\tilde{u}_i$ as the integer that has u_i as binary

representation and $\tilde{v}_i$ as the integer that has v_i as binary representation. We build a counter system with two states such that a run on this system corresponds to a sequence of elements of D, by encoding in binary the concatenation of the two words obtained with the sequence into the two counters. Note that the 0 at the beginning of the words should not be omitted in the counter value. To do this, we add a bit at one at the beginning of the binary encodings.

Let $Q = \{q, \bot\}$, let T be the set of edges, which contains all $t_i := (q, (2^{h_i}x + \tilde{u}_i, 2^{k_i}x + \tilde{v}_i), q)$ for $1 \leq i \leq n$ and two additional edges $(q, (x-1, x-1), \bot)$ and $(\bot, (x-1, x-1), \bot)$.

Let us have a look at the binary encoding of the counters. The edge t_i pads the two counters by the number of bits that corresponds to the size of the words u_i and v_i, respectively, then adds the integer that is represented by these words to the first and second counter, respectively, which fills the padded bits, possibly with 0 at the beginning if they are indeed in the words.

In the counter system (Q, T), there is a run from $(q, (1, 1))$ to $(\bot, (1, 1))$ if, and only if, the instance D of the PCP problem is positive.

Indeed, let $i_1, i_2, \ldots, i_k$, with $k > 0$, be a sequence of indices in $\{1, \ldots, n\}$ such that $u_{i_1}u_{i_2}\ldots u_{i_k} = v_{i_1}v_{i_2}\ldots v_{i_k}$.

Consider a run that takes the edges $t_{i_1}, t_{i_2}, \ldots, t_{i_k}$ in this order. After these k steps, the two counter values are equal and their binary encoding, which we identify to a word on $\{0, 1\}$, is $1u_{i_1}u_{i_2}\ldots u_{i_k}$. Let v be the common counter value. By taking the edge $(q, (x-1, x-1), \bot)$ and $v - 2$ times the self-loop on $\bot$, the run reaches $(\bot, (1, 1))$.

For the other implication, if there is a run from $(q, (1, 1))$ to $(\bot, (1, 1))$, then such a run reaches necessarily a configuration $(q, (y, y))$ where $y > 2$. Yet, the configuration can only be reached by taking a sequence of transitions t_i, for $1 \leq i \leq n$. According to our remark about the effect of these transitions, the sequence of indices of the transitions that the run takes gives a witness for the instance D of the PCP problem. □

2.6 Hierarchical counter systems

In this section, we consider a model of counter systems inspired by the *Vector Addition Systems (VAS) with hierarchical zero-tests*, studied in [Rei08], and on the *multi-pushdown automata* (MPDA) defined in [BCCCR96].

VAS with hierarchical zero-tests have been introduced in the unpublished work [Rei95] by Klaus Reinhardt. They extend VAS with one zero-test, in which only one counter can be tested to zero, in the following way: the counters are numbered, and whenever one of them is tested to zero, all others with a lower index are tested to zero as well. Note that allowing even that two counters can be tested to zero without restriction leads to a Turing-powerful model. According to the usual equivalence between VAS and VASS, a VASS with hierarchical zero-tests is defined as a VASS such that there exist special transitions that can be fired from their starting vertex if, and only if, the

first k counters are all zero, where k is a positive integer smaller than or equal to the dimension of the VASS.

Multi-pushdown automata are pushdown automata with multiple stacks, which are linearly ordered, with transitions that depend on the configuration: In a run on an MPDA, the pop operation can only be performed on the first non-empty stack.

According to [Ati10], model-checking ω-regular properties for MPDA is 2-EXPTIME-complete, a result that extends the 2-EXPTIME-completeness of the emptiness problem for MPDA in [ABH08]. In the latter article, the authors reduce another model, bounded-phase multi-stack pushdown automata, to MPDA with twice as many stacks. In bounded-phase multi-stack pushdown automata, the phase is a part of a run in which all operations affect only one stack, and all runs must have a bounded number of phases. Anil Seth studied parity games on bounded-phase multi-stack pushdown automata in [Set09] and the reachability problem on the same model in [Set10].

The comparison between pushdown automata and counter systems shows some similarities. A stack with an alphabet that contains a single letter is equivalent to a natural integer, and push/pop represent incrementation/decrementation. Also, a stack with an alphabet that contains two letters can represent a relative integer, which is the difference between the number of occurrences of one of the letters and the number of occurrences of the other one. It is by the way useful that the stack never contains the two letters at the same time, so that the size of the stack is the absolute value of the integer that the stack represents, whereas the sign corresponds to the letter that the stack contains.

A natural translation of multi-pushdown automata to counter systems is the following: In a run on a counter system, the i^{th} counter can only be decreased if the counters numbered from 1 to $i-1$ are all zero. We may imagine another translation by replacing the last word "zero" by "nonpositive".

Actually, the model that we define is slightly different, and we name it *hierarchical counter system*. In a run on a hierarchical counter system, the i^{th} counter can only be *modified* if the counters numbered from 1 to $i-1$ are all zero. We also suppose that the vector in the label of all edges has at most one non-zero component. This is without loss of generality because it is possible to split for example an edge with label $(4,2,3,0)$ into the three consecutive edges with labels $(0,0,3,0)$, $(0,2,0,0)$ and $(4,0,0,0)$, in this very order. The chain of edges is enabled in a run if, and only if, the sum is enabled.

We consider hierarchical counter systems under the $\mathbb{Z}$ semantics, but the semantics itself has no influence on the proof: we can use the same proof for the other two semantics that we defined in this chapter. Note that hierarchical counter systems under the VASS semantics are a particular model of VASS with hierarchical zero-tests. Indeed, to forbid the update of a counter when the lower counters are not zero is equivalent to have transitions that tests the first k counters for zero before any transition that updates the $(k+1)^{\text{th}}$ counter.

2.6.1 The reachability problem on hierarchical counter systems

The reachability problem on hierarchical counter systems is the following: Given a hierarchical counter system (Q,T) in dimension d, a vector $(c_1,\dots,c_d)$ and two vertices q and q_f, decide whether there is a run from $(q,(c_1,\dots,c_d))$ to $(q_f,(0,\dots,0))$. Note that, even though there is no translation invariance in hierarchical counter systems, i.e., there can be a run from (q,x) to (q',x') but no run from $(q,x+y)$ to $(q',x'+y)$ where $q,q'\in Q$ and $x,x',y\in\mathbb{N}^d$, the complexity would not be affected if we had $(0,\dots,0)$ as initial counter vector or an arbitrary counter vector in the objective. The reachability problem on VAS with hierarchical zero-tests, defined similarly but with $(c_1,\dots,c_d)=(0,\dots,0)$, is decidable according to [Rei08], which guarantees that the reachability problem on hierarchical counter systems under the VASS semantics is decidable too, and we prove in this section that the problem is in NP under any semantics.

Intuitively, in hierarchical counter systems, the update of a counter at zero freezes all edges that modify counters of higher numbers. We can thus consider that there are as many underlying counter systems as counters, the counter system number i is obtained by removing edges that modify a counter with a number $> i$.

Proposition 23: *The reachability problem on hierarchical counter systems is in NP.*

Proof. Let (Q,T) be a hierarchical counter system in dimension d, let $(c_1,\dots,c_d)$ be a counter vector, let q and q_f be two vertices. Because we imposed that edges modify one counter only, we here redefine T as a subset of $Q\times\mathbb{Z}\times\{1,\dots,d\}\times Q$. With this redefinition of T, when an edge (q,k,i,q') is taken from a configuration $(q,(x_1,\dots,x_d))$, first, it is required that $x_1=\dots=x_{i-1}=0$, and second, the next configuration is $(q',(\underbrace{x_1,\dots,x_{i-1}}_{\text{all zero}},x_i+k,x_{i+1},\dots,x_d)$.

Note that on any run from $(q,(c_1,\dots,c_d))$ to $(q_f,(0,\dots,0))$, if it exists, the following counter vectors are necessarily reached in this order: $(0,c_2,\dots,c_d),(0,0,c_3\dots,c_d),\dots,(0,\dots,0,c_d)$. However, it is possible that for example between reaching $(0,c_2\dots,c_d)$ and $(0,0,c_3,\dots,c_d)$ the run reaches a counter vector with a non-zero value in the first counter. If some c_i are zero, then the list of configurations contains multiple entries, but anyway counter vectors that appear more than once may be reached once only.

Let us first describe a naive procedure to solve the problem. This procedure has the drawback of computing everything that could be useful, without making any guess. This leads to a complexity in the class P^{NP}, which is the class, also called Δ_2^{P}, of problems decided by a deterministic Turing machine halting in polynomial time and that can use an oracle to solve any problem in NP in a single operation. This class is in the polynomial hierarchy, and the polynomial hierarchy is included in PSPACE, as proved in [MS72].

For $0\le i\le d$, we denote by $T_{\le i}$ the subset of T that contains exactly the edges that modify a counter $\le i$, in other words $T_{\le i}=T\cap(Q\times\mathbb{Z}\times\{1,\dots,i\}\times Q)$. In particular, $T_{\le d}=T$ and $T_0=\emptyset$. We compute for all $1\le i\le d$ the pairs of vertices (r,s) such that there exists a run from

$(r,0)$ to $(s,0)$, with edges in $T_{\leq i}$ only.

For $1 \leq i \leq d$, we denote by $T_{=i} \in Q \times \mathbb{Z} \times Q$ the projection on the i^{th} counter of $T_{\leq i} \setminus T_{\leq i-1}$, i.e., $T_{=i} = \{(r,v,s) \mid (r,v,i,s) \in T\}$, and we build by induction the sets $Q_i \subseteq Q$, $W_i \subseteq Q \times Q$ and $T_i \subseteq Q \times \mathbb{Z} \times Q$. We obtain i one-dimensional counter systems (Q,T_i) on which we call a solver of the reachability problem. Until now, the semantics played no role, and its only influence is that the systems (Q,T_i) are under the same semantics as (Q,T).

For the naive procedure, the induction starts with $T_1 := T_{=1}$. We compute successively:

- the set Q_1 of vertices s such that there exists a run in (Q,T_1) from (q,c_1) to $(s,0)$;
- the relation $W_1 \subseteq Q \times Q$ such that $(r,s) \in W_1$ if, and only if, there exists a run in (Q,T_1) from $(r,0)$ to $(s,0)$;
- for $2 \leq i \leq d-1$:
  - the set $T_i := T_{=i} \cup \{(r,0,s) \mid (r,s) \in W_{i-1}\}$;
  - the set Q_i of vertices s such that there exists a vertex $r \in Q_{i-1}$ such that there exists a run in (Q,T_i) from (r,c_i) to $(s,0)$;
  - the relation $W_i \subseteq Q \times Q$ such that $(r,s) \in W_i$ if, and only if, there exists a run in (Q,T_i) from $(r,0)$ to $(s,0)$;
- the set $T_d := T_{=d} \cup \{(r,0,s) \mid (r,s) \in W_{d-1}\}$;
- the set Q_d of vertices s such that there exists a vertex $r \in Q_{d-1}$ such that there exists a run in (Q,T_d) from (r,c_d) to $(s,0)$;

We claim that there exists a run from $(q,(c_1,\ldots,c_d))$ to $(q_f,(0,\ldots,0))$ if, and only if, $q_f \in Q_d$. Indeed, for all $1 \leq i \leq d$, the set T_i gathers all edges that either modify the i^{th} counter (the subset $T_{=i}$) or replace an existing run ρ in T_{i-1} with labels that sum up to zero (the subset based on W_{i-1}, for "warping edges"), and edges in ρ can themselves already replace runs in previous sets. Accordingly, for all $1 \leq i \leq d$, the set Q_i gathers all vertices s such that there exists a run from $(q,(c_1,\ldots,c_d))$ to $(s,(0,\ldots,0,c_{i+1},\ldots,c_d)$ in $(Q,T_{\leq i})$.

Here, the number of calls to an oracle that solves the NP-complete reachability problem on a one-dimensional counter system (see Theorem 6 on page 26, [HKOW09] or Theorem 13 on page 36 depending on the semantics) is $d \cdot |Q|^2 + d$ and the counter systems (Q,T_i) have the size of (Q,T), which leads to the announced complexity.

Now, in a second time, we use the same sets, but we do not compute them. Instead, everything is a matter of guessing at each step elements or pairs of elements of Q, checking that they are indeed in the Q_i or in a relation W_i or T_i, respectively.

More precisely, we guess checkpoints of the run of which we want to prove the existence. In other words:

- guess $s_1 \in Q$, check that there exists a run in (Q, T_1) from (q, c_1) to $(s_1, 0)$;
- guess elements (r, s) of $Q \times Q$ that are in W_1, call W_1' their collection, check that there exists a run in (Q, T_1) from $(r, 0)$ to $(s, 0)$ for each $(r, s) \in W_1'$;
- for $2 \leq i \leq d-1$:
  - define T_i' as $T_{=i} \cup \{(r, 0, s) \mid (r, s) \in W_{i-1}'\}$, which is a subset of T_2 that contains required edges;
  - guess $s_i \in Q$, check that there exists a run in (Q, T_i') from (s_{i-1}, c_i) to $(s_i, 0)$;
  - guess elements (r, s) of $Q \times Q$ that are in W_i, call W_i' their collection, check that there exists a run in (Q, T_i) from $(r, 0)$ to $(s, 0)$ for each $(r, s) \in W_i'$;
- define T_d' as $T_{=d} \cup \{(r, 0, s) \mid (r, s) \in W_{d-1}'\}$;
- check that there exists a run in (Q, T_d') from (s_{d-1}, c_d) to $(q_f, 0)$.

Even though we do not consider some edges based on sets W_i, it is not necessary to guess that they exist if the run does not cross them. Hence, there is a polynomial number of guesses, at most $d \cdot |Q|^2 + d$, and each guess corresponds to a call to a solver of the reachability problem in dimension one. Now, because we need to check that the guesses are correct, the calls to a solver require an answer yes, hence that the instance is positive, and the whole procedure answers no else. This guarantees that the procedure that we present solves the reachability problem on (Q, T) in nondeterministic polynomial time. □

Note that the notion of hierarchical counter systems is irrelevant in dimension one, hence a lower complexity bound for hierarchical counter systems under any semantics in any dimension is the lower complexity bound under the same semantics in dimension one.

2.6.2 Reachability games on hierarchical counter systems

In the heuristic for solving the reachability problem on hierarchical counter systems, a crucial point is that we compute the set of pairs of vertices such that there exists a path from one to another with labels that sum up to zero. In reachability games, it is unlikely to have a set of pairs of vertices (q, r) such that Eve has a strategy to go from (q, x) to (r, x) for any $x \in \mathbb{N}$, for example in the simple arena given in Figure 2.20, whatever happens, Eve has a strategy to go from $(q_0, 1)$ to a configuration $(q, 1)$, where $q \in \{q_3, q_4, q_5, q_6\}$, but she does not have a strategy to go to a particular vertex of the set, it depends on Adam's first move.

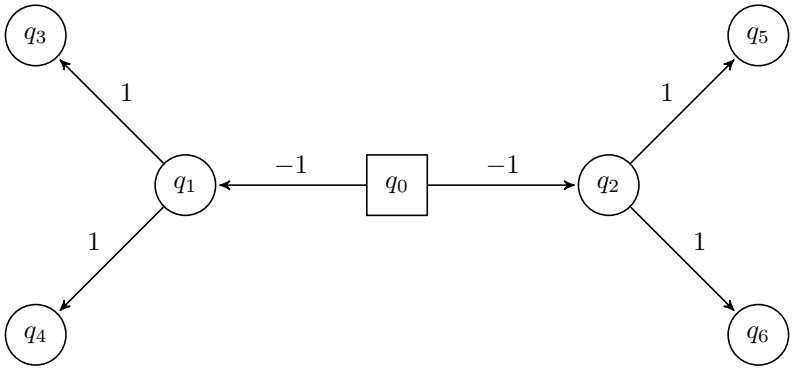

Figure 2.20 – Counter reachability game on which Eve has no winning strategy to reach a particular vertex.

2.7 Discussion

In this chapter, using unary systems was crucial and many results needed such systems. Actually, it represents the fact that inputs, in particular counter values in the labels, are encoded in unary. With binary encodings, the complexity of our algorithms becomes exponential, yet we can find algorithms of lower complexity in a few cases thanks to properties of some systems. For example, in the section about non-blocking VASS, Algorithm 1 uses polynomial time only when the initial counter value, which is in input, is zero, and we use it precisely in this case, while we take care of other values in a different way, namely as we solve a counter reachability game on the VASS that is formed by a subgraph of the counter system, with objective zero in any vertex, like in [BJK10].

When a counter system is not unary, determining the winner of a reachability game is EXPSPACE-complete, according to the article [Hun14]. In this article, the author presents a model of one-dimensional counter systems with which we can cover all three semantics, and we can even include reset counter systems. There are in fact three types of edges: regular edges, edges that are activated when the counter is zero and edges that are activated when the counter is in $\mathbb{Z}\backslash\{0\}$. Edges of the last two types have label zero. The EXPSPACE lower bound is established by two consecutive reductions: from the EXPSPACE-complete CTL model checking on succinct one-counter automata to Büchi games on one-counter graphs and from these Büchi games to counter reachability games.

Chapter 3

An Example of Counter Reachability Games: Robot Games

Robot games, introduced in [DR13] are counter reachability games on an arena with only one vertex for each player. There is no self-loop, hence the plays proceed in rounds. In Figure 3.1, we represent a robot game by drawing the arena, and another possible representation consists of giving the set A of labels of moves from Adam's vertex to Eve's vertex and the set E of labels of moves from Eve's vertex to Adam's vertex.

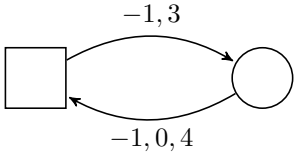

Figure 3.1 – Example of a robot game for the sets $A = \{-1, 3\}$ and $E = \{-1, 0, 4\}$.

The winning condition for Eve in a robot game is that the counter vector is zero after her turn. We decide that Adam always starts, and players have perfect information, so they know the value of the counter vector before they play.

The first mention of robot games shows an exemple in dimension two, where determining whether Eve has a winning strategy is given as an open problem, unlike most counter reachability games for which undecidability has been proved. The study of robot games, though, was most fruitful and motivated in dimension one, where the complexity of the problem of determining the winner was settled, unlike, at that time, the lower complexity bound of the same problem for counter reachability games in dimension one.

We give in this chapter an EXPTIME algorithm for determining the winner in dimension one,

then we prove that the complexity is optimal by a reduction of the decision problem associated with countdown games, presented in [JLS07]. After that, we prove undecidability of robot games in dimension three in three steps.

Definitions and basic properties

A *robot game* in dimension d is a pair (A, E), where A and E are finite subsets of $\mathbb{Z}^d$. The robot game is played by Adam, who owns the subset A, and Eve, who owns the subset E. Given an initial counter vector in $\mathbb{Z}^d$, a *play* proceeds in *rounds*. In a round that starts at the counter vector $x \in \mathbb{Z}^d$, Adam chooses what we call a *move* $a \in A$ and updates the counter vector to $x + a$, then Eve chooses a move $e \in E$ and updates the counter vector to $x + a + e$, in which the round ends. Eve wins and the play stops if the counter vector is zero when a round ends, else a new round is played. By convention, Eve wins immediately when a play starts with zero as counter vector.

The notion of *strategy* is inherited from counter reachability game. Here, only the counter value just before the turn of the player is important, not its evolution before. In fact, if we consider the robot game as a reachability game on the infinite arena $\{\Box, \bigcirc\} \times \mathbb{Z}^d$, with objective $(\Box, (0))$, the latter game is determined according to the Gale-Stewart theorem in [GS53]. We use again this comparison when we introduce the attractor.

A counter vector x is *winning* if there exists a strategy of Eve, such that for all strategies of Adam, Eve wins the play that starts with x and in which each player moves according to his strategy. We switch Eve and Adam in the last sentence to define the notion of a *losing* counter vector. The decision problem associated with a robot game (A, E) and an initial counter vector $x \in \mathbb{Z}^d$ asks whether x is winning.

In dimension one, the *amplitude* of a robot game (A, E) is the integer interval bounded by the extremal combinations of moves in a round. We denote it by $\mathrm{Ampl}(A, E) = [\![\min(A) + \min(E), \max(A) + \max(E)]\!]$. We also define for any $k \in \mathbb{N}$ the integer interval $\mathrm{Ampl}^k(A, E) = [\![\min(A) + \min(E) - k, \max(A) + \max(E) + k]\!]$.

We now give some basic properties of robot games. Let us first remark that robot games are translation invariant: Whenever a player can make a move and change the counter vector from x to x', the same move changes the counter vector from y to $y - x + x'$.

Proposition 24: *If two counter vectors are winning in a robot game, then their sum is also winning.*

Proof. Let $x \in \mathbb{Z}^d$ and $y \in \mathbb{Z}^d$ be two winning counter vectors. Let σ_x and σ_y be winning strategies of Eve from x and y. Because the game is translation invariant, in a play that starts with $x + y$, Eve can enforce a round to end with counter vector y, with the strategy $z \mapsto \sigma_x(z - y)$. After such a round, Eve wins by using σ_y. □

As a consequence, if all counter vectors in a set $X \subseteq \mathbb{Z}^d$ are winning in a robot game, then every X-reachable counter vector is winning. This guarantees that the winning set is closed under addition and multiplication by a nonnegative integer.

The next proposition states what happens in dimension one when a player can force the counter value to increase or decrease unboundedly.

Proposition 25: *Let (A, E) be a robot game in dimension one.*

- *If $\max(A) \geq -\min(E)$, then each positive counter value is losing. Similarly, if $\min(A) \leq -\max(E)$, then each negative counter value is losing.*
- *If $\max(E) > -\min(A)$, and if there exists a bound above which each counter value is winning, then each counter value is winning. The same holds if $\min(A) < -\max(E)$, and if there exists a bound below which each counter value is winning.*

Proof.

- We consider a robot game (A, E) in which we have $\max(A) \geq -\min(E)$. For any positive counter value x and all moves $a_1, \ldots, a_k \in A$, $e_1, \ldots, e_k \in E$, Adam wins by playing always $\max(A)$, because every round ends with a counter value that is greater than or equal to the one of the previous round, no matter what Eve does. The case where $\min(A) \leq -\max(E)$ is analogous for negative counter values.
- (First case only, the second one is analogous) We consider a robot game (A, E) for which we have $\max(E) > -\min(A)$, and for any counter value y above a certain $x \in \mathbb{Z}$, Eve has a winning strategy σ_y. Here is a possible winning strategy for Eve from any initial counter value: In a play prefix where no round ended with a counter value above x, she plays $\max(E)$; if the counter value is z and the first time when a round ended with a counter value above x, this value was y, she plays $\sigma_y(z)$. Because $\max(E) > -\min(A)$, the counter value grows after every round until it goes over x and Eve will win afterwards. Like in the proof of Proposition 24, Eve knows whether a round already ended with a counter value $y > x$ and she knows in that case the value y.

□

The first result of Proposition 25 extends well in dimension d: if, for any $1 \leq k \leq d$, the greatest component along dimension k of a vector in A is greater than the opposite of the smallest component along dimension k of a vector in E, then each counter vector with a positive component along dimension k is losing. Conversely, if, for any $1 \leq k \leq d$, the smallest component along dimension k of a vector in A is smaller than the opposite of the greatest component along dimension k of a vector in E, then each counter vector with a negative component along dimension k is losing.

3.1 Algorithm for determining the winner in dimension one

In this section, we present the tools to build an exponential-time algorithm that determines the winner in a robot game. First, we explain the notion of an attractor, then we define the Frobenius problem and we give two over-approximations of the solution to this problem, in order to find bounds above and below which we are sure that the same player always wins. The algorithm in the fourth subsection computes the attractor and uses the bounds that we get to avoid infinite recursion.

3.1.1 The attractor construction

The attractor construction is a well known tool to determine the winner of a reachability game. It is for example presented in [GTW02, Section 2.5]. We use different notations in our work.

We first define the one-step attractor of a set. Consider a graph (Q, Q_E, Q_A, T) for a general reachability game, where Q is a possibly infinite set of vertices partitioned into subsets Q_E for Eve and Q_A for Adam, and $T \subseteq Q \times Q$. The *one-step attractor* of a subset X of Q, here written $\mathrm{Attr}(X)$, is the set of states from which Eve can force to go to X in one step, which means:

$$\begin{aligned}\mathrm{Attr}(X) = &\{q \in Q_E \text{ such that } \exists q' \in X, (q, q') \in T\} \\ &\cup \{q \in Q_A \text{ such that } \forall q' \in Q, (q, q') \in T \text{ implies } q' \in X\}.\end{aligned}$$

The *attractor* of X, written $\mathrm{Attr}^*(X)$, is the set of states from which Eve has a strategy to eventually go to X no matter what Adam plays, in other words she has a winning strategy in the reachability game with objective X on the aforementioned arena. The set $\mathrm{Attr}^*(X)$ is the least fixpoint of Attr containing X. We obtain it recursively: compute $Y = X \cup \mathrm{Attr}(X)$, if $Y = X$ then return Y else set $X := Y$ and repeat.

Let us adapt a robot game to these notations. Eve owns $Q_E := \{\bigcirc\} \times \mathbb{Z}$ and Adam owns $Q_A := \{\square\} \times \mathbb{Z}$. The set of edges is the union of the set $\{((\square, x), (\bigcirc, y)) \mid x, y \in \mathbb{Z}, y - x \in A\}$, which represents the moves of Adam, and of the set $\{((\bigcirc, x), (\square, y)) \mid x, y \in \mathbb{Z}, y - x \in E\}$, which represents the moves of Eve. The objective for Eve is the vertex $(\square, 0)$. In our definition of robot games, winning positions are counter values. They are here represented as a pair ($\square/\bigcirc$, the counter value), but we only care for winning positions with $\square$ as left component when we solve the game, because a play starts with Adam.

We use here two-step attractors $\mathrm{Attr}^2(X) = \mathrm{Attr}(\mathrm{Attr}(X))$, rather than one-step attractors, because of the round-based structure of a play in the robot game. The winning set in a robot game is $\mathrm{Attr}^*(\{(\square, 0)\})$. We call it trivial if its intersection with the set of vertices of Adam is restricted to $\{(\square, 0)\}$, which is the case if, and only if, the computation of $\mathrm{Attr}^*(\{(\square, 0)\})$ stops at the second step because a fixpoint has already been reached. In other words, the winning set in a robot game is trivial if, and only if, the set $\mathrm{Attr}^2(\{(\square, 0)\})$ is $\{(\square, 0)\}$.

Proposition 26: *The winning set in a robot game (A, E) is non-trivial if, and only if, there exists a counter value $x \neq 0$ such that, for all moves of Adam $a \in A$, there exists a move $e \in E$ of Eve such that $a + e = -x$.*

Proof. Given a counter value $x \neq 0$, a configuration $(\square, x)$ is in $\mathrm{Attr}^2(\{(\square, 0)\})$ if, and only if, for any move $a \in A$ of Adam, we have $(\bigcirc, x+a) \in \mathrm{Attr}(\{(\square, 0)\}$, and this is equivalent to the existence of an $e \in E$, which depends on a and x, such that $x + a + e = 0$. □

Let us look at the game presented in the Figure 3.1. Here, consider a play that starts at -3. If Adam chooses to play 3, then Eve wins by playing 0, if Adam plays -1, then Eve wins by playing 4. With the terminology of this section, it means that $(\square, -3) \in \mathrm{Attr}^2(\{(\square, 0)\})$.

We define an integer version of Attr^2, for a subset X of $\mathbb{Z}$, by

$$\mathrm{Pre}(X) = \{x \in \mathbb{Z} \mid (\forall a \in A)(\exists e \in E)\ x + a + e \in X\}.$$

Note that a round begins in $\mathrm{Pre}(X)$ if, and only if, Eve can force this round to end in X. Let X be a subset of $\mathbb{Z}$, and let $X_{\mathrm{opp}} = \{\square\} \times X$. Because the predecessor of a vertex with $\square$ in the left component can only be a vertex with $\bigcirc$ in the left component and vice-versa, we have $\mathrm{Attr}^2(X_{\mathrm{opp}}) = \{\square\} \times \mathrm{Pre}(X)$.

The next result is very important for our algorithm. We will usually be in a situation where the algorithm computes a bound b such that we can decide immediately for which player a counter value x that is greater in absolute value than b is winning. The proposition presents the bounded arena that we build from the robot game, where termination is guaranteed for the computation of the attractor.

Proposition 27: *Consider a robot game G for which there exist two integers $d \in \mathbb{N} \setminus \{0\}$ and $b \in \mathbb{N}$ such that no negative counter value is winning and every counter value greater than b is winning if, and only if, it is a multiple of d. We can build a reachability game on a finite arena on which Eve has a winning strategy if, and only if, she has a winning strategy in G.*

Proof. Let $\mathrm{Restr}^b_d(A, E) = (Q, Q_E, Q_A, T)$, where:

- $Q_A = \{\bot_{<0}, \top_{>b}, \bot_{>b}\} \cup (\{\square\} \times [\![0, b]\!])$;
- $Q_E = \{\bigcirc\} \times [\![\min(A), b + \max(A)]\!]$;
- $Q = Q_E \cup Q_A$;
- $T = \{((\square, x), (\bigcirc, y)) \in Q_A \times Q_E \mid y - x \in A\}$
  $\cup\, \{((\bigcirc, x), (\square, y)) \in Q_E \times Q_A \mid y - x \in E\}$
  $\cup\, \{((\bigcirc, x), \bot_{<0}) \in Q_E \times Q_A \mid \exists e \in E, x + e < 0\}$
  $\cup\, \{((\bigcirc, x), \bot_{>b}) \in Q_E \times Q_A \mid \exists e \in E, x + e > b \wedge x + e \notin d\mathbb{Z}\}$

$\cup \{((\bigcirc, x), \top_{>b}) \in Q_E \times Q_A \mid \exists e \in E, x + e > b \wedge x + e \in d\mathbb{Z}\}$
$\cup \{(\bot_{<0}, \bot_{<0}), (\top_{>b}, (\square, 0)), (\bot_{>b}, \bot_{>b})\}$.

The reachability game played on $\text{Restr}_d^b(A, E)$ where Eve's objective is to go from $(\square, x)$ to $(\square, 0)$, for a given $1 \leq x \leq b$, is actually the robot game (A, E) with the initial value x, in which the play stops as soon as the winner is decided. Indeed, we supposed that all negative counter values are losing, that is why, on $\text{Restr}_d^b(A, E)$, instead of vertices (Q_A, x) for $x \in -\mathbb{N}$ we have a losing sink $\bot_{<0}$. Similarly, because in (A, E) all counter values above b are winning if, and only if, they are multiples of b, on $\text{Restr}_d^b(A, E)$, instead of vertices (Q_A, x) for $x > b$, we have two sinks, one winning and one losing, and the redirection of edges depends on the counter value. □

The notation $\text{Restr}_d^b(A, E)$ is extended to negative integers b whenever no positive counter value is winning and every counter value less than b is winning if, and only if, it is a multiple of d. In fact, $\text{Restr}_d^b(A, E)$ is the arena $Restr_d^{-b}(-A, -E)$. Given d and b, we can decide the winner in the reachability game on $\text{Restr}_d^b(A, E)$ using the attractor construction, because this time the arena is finite. We write $\text{RestrAttr}(G) = \{x \in \mathbb{Z} \mid (\square, x) \in \text{Attr}^*(\{(\square, 0)\})\}$ where $\text{Attr}^*(\{(\square, 0)\})$ is the winning set in the game described above on the arena $G = \text{Restr}_d^b(A, E)$. The function RestrAttr is used in the main algorithm.

3.1.2 The Frobenius problem

Let W be a non-empty subset of $\mathbb{Z}$. The arithmetical notions that we present in this section are part of the algorithm: W stands for a subset of the winning counter values. We denote by $\gcd(W)$ the greatest common divisor of W, which we compute as follows: $\gcd(\{d\}) = d$, and for $W \neq \emptyset$, $\gcd(\{w\} \cup W)$ is the usual greatest common divisor of w and $\gcd(W)$. The integers in W are mutually prime if $\gcd(W) = 1$.

The *Frobenius problem* asks for the greatest integer that is not W-reachable, where W is a set of mutually prime positive integers.

Note that the set of non-W-reachable positive integers would be infinite without the assumption of mutual primality. It is empty whenever the set W contains the value 1, in which case the solution to the Frobenius problem is -1, by convention. Theorem 28, which follows from [Wil78], gives a bound to the solution to the Frobenius problem for a given set.

Theorem 28 ([Wil78])**:** *Let W be a set of mutually prime positive integers. The solution to the Frobenius problem for W is less than or equal to* $\max(W)^2$.

Here, we are interested in a variant of the Frobenius problem on arbitrary subsets W of $\mathbb{N}$ or $-\mathbb{N}$, where we look for a bound beyond which every integer is W-reachable if, and only if, it is a multiple of $\gcd(W)$. When W is a set of mutually prime positive integers, this is exactly the Frobenius problem. Otherwise, let $W \subseteq \mathbb{N}$ and $d = \gcd(W)$. Consider the set $W' = \{\frac{w}{d} \mid w \in W\}$, which contains mutually prime positive integers. Let F be the solution to the Frobenius problem

for W'. The set of W-reachable integers greater than dF is equal to the set of multiples of d greater than dF. Consequently, the solution to our problem for W is dF. For $W \subseteq -\mathbb{N}$, we procede analogously.

Actually, the computation of F is hard and the bound in Theorem 28 has at most twice the size of W. That is why we use the following function in the algorithm. Let $W \subseteq \mathbb{N}$ (resp. $-\mathbb{N}$) such that $\gcd(W) = 1$. We write $\tilde{F}(W) = \max(W)^2$ (resp. $-\max(-W)^2$, which is also $-\min(W)^2$), beyond which all integers are W-reachable. We extend $\tilde{F}(W)$ when $\gcd(W) = d \neq 1$: Let $W' = \{\frac{w}{d} \mid w \in W\}$, we set $\tilde{F}(W) := d\tilde{F}(W') = \frac{\max(|W|)^2}{d}$. In the particular case where W is a singleton, we set $\tilde{F}(W) = 0$.

Finally, when W is neither included in $\mathbb{N}$ nor in $-\mathbb{N}$, we decide W-reachability according to the following lemma.

Lemma 29: *Let W be a finite subset of $\mathbb{Z}$ that has two elements of opposite signs. An integer is W-reachable if, and only if, it is a multiple of gcd(W).*

Proof. Consider two elements $w > 0$ and $w' < 0$ of W. The integers $-w$ and $-w'$ are W-reachable because $-w = (-w' - 1)w + ww'$ and $-w' = (w - 1)w' + (-w')w$, which are combinations with only nonnegative coefficients.

By Bézout's identity, there exist integer coefficients a_w such that $\sum_{w \in W} a_w w = \gcd(W)$. We replace $a_w w$ by $(-a_w) \cdot (-w)$ for all negative coefficients a_w. The resulting linear combination has only positive coefficients, therefore $\gcd(W)$ is W-reachable, as well as $-\gcd(W)$. We conclude that W-reachability is equivalent to membership in $\gcd(W)\mathbb{Z}$. □

3.1.3 A theorem by Sylvester

This section aims at giving an alternative way to bound the solution to the Frobenius problem. Using Theorem 28 is simpler, but we can have a sharper bound.

Theorem 30 relies on the extended Euclidean algorithm, presented in [CLRS09, p. 937]. With an iteration of the algorithm to more than two integers according to the way we present the greatest common divisor of a set, we can prove Corollary 31.

Theorem 30: *Let a, b be two integers. Bézout coefficients for a and b, that is to say integers u, v such that $ua + vb = gcd(a, b)$, can be computed with a time complexity polynomial in the size of the binary encoding of a and b.*

Corollary 31: *Bézout coefficients for a finite subset of $\mathbb{Z}$ are computable with a polynomial time complexity in the size of the binary encoding of the integers in the subset.*

The article [PRS05] mentions a theorem concerning the Frobenius problem, due to Sylvester

in [Syl82].

Theorem 32 ([Syl82])**:** *Let $W = \{p, q\}$ be an instance of the Frobenius problem. Then the greatest non-W-reachable integer is $pq-p-q$. Moreover, an integer x is reachable if, and only if, $pq-p-q-x$ is not, which means exactly half of the integers between 0 and $pq - p - q$ are reachable.*

Such a simple statement does not extend well when W has more than two elements. If there are two mutually prime integers in W, then we can take them, else we have to find two mutually prime W-reachable integers to get an upper bound of the maximal non-W-reachable integer. For example, if $W = \{6, 10, 15\}$, then there is no pair of mutually prime integers in W even though $\gcd(W) = 1$. Nevertheless, 25 is W-reachable and we have $\gcd(6, 25) = 1$, hence every integer above $6 \cdot 25 - 6 - 25$ is W-reachable. In any case, Proposition 33 gives a way to apply Theorem 32 and there is only a finite number of integers for which we cannot find out immediately whether they are W-reachable or not.

Proposition 33: *Let W be a subset of $\mathbb{Z}$ such that $gcd(W) = 1$. There is a polynomial time algorithm that gives a pair of mutually prime W-reachable integers.*

Proof. If two integers in W are mutually prime, then there is nothing to do. Else, consider the linear combination, obtained with the extended Euclidean algorithm, $\sum_{i=1}^{n} a_i w_i = 1$ for $w_1, \ldots, w_n \in W$. We suppose that the terms are ordered such that for a certain $1 \leq k \leq n+1$ all $a_i, i < k$ are positive and all $a_i, i \geq k$ are negative. Note that $n \geq 3$, else two integers in W would be mutually prime. Let us distinguish three cases:

- If $k = 1$, in other words all a_i are negative, then consider the W-reachable integers $p := w_1$ and $q := \sum_{i=2}^{n}(-a_i)w_i$. We apply Bézout's theorem: p and q are mutually prime because $a_1 p - q = 1$.

- Similarly, if $k = n+1$, in other words all a_i are positive, then the W-reachable integers $p := w_1$ and $q := \sum_{i=2}^{n} a_i w_i$ are mutually prime.

- Else, the W-reachable integers $p := \sum_{i=1}^{k-1} a_i w_i$ and $q := \sum_{i=k}^{n} -a_i w_i$ are mutually prime and defined by a non-empty sum.

□

3.1.4 The algorithm

We have now all necessary tools to solve robot games. The main idea is to iterate the computation of Pre until we establish that we can describe the winning set with the finite set obtained so far.

Let X be the set that we compute in the first step of our algorithm and Win the winning set in the robot game. We prove in Proposition 34 and its corollary that $X \subseteq$ Win and that, for

a well-chosen set Y' of X-reachable counter values, if $\gcd(\mathrm{Pre}(Y')) = \gcd(X)$, then also $\gcd(\mathrm{Win}) = \gcd(X)$. Basically, the first step relies on this property: We compute successive Pre, and once the step ends we get $\gcd(\mathrm{Win})$. Actually, to keep control over the complexity, we do not do $X := X \cup \mathrm{Pre}(X)$, but only add to X a single element y of the computed Pre such that $\gcd(X) \neq \gcd(X \cup \{y\})$.

Once we find $\gcd(\mathrm{Win})$, there are two cases. In the first case, Lemma 29 can be applied and we are done, because Win has two elements with opposite signs. Hence, the winning set is $\gcd(\mathrm{Win})\mathbb{Z}$. In the second case, the winning set is included in one of the sets $\mathbb{N}$ or $-\mathbb{N}$; we suppose without loss of generality that the winning set is included in $\mathbb{N}$. Theorem 28 yields a bound above which all multiples of $\gcd(\mathrm{Win})$ and only them are winning because they are X-reachable. Therefore, the only set of counter values about which we still do not know whether they are winning or not is empty or bounded and, by Proposition 27, we can compute an attractor on the restricted arena.

Proposition 34: *Let Win be the winning set in a robot game (A, E), and $d \in \mathbb{N}$ be a multiple of gcd(Win) that is not gcd(Win). Let $Y = d\mathbb{Z} \cap \mathrm{Ampl}^d(A, E)$. Then we have $\mathrm{Pre}(Y) \setminus d\mathbb{Z} \neq \emptyset$.*

Proof. First, we prove that if $\mathrm{Pre}(d\mathbb{Z})$ is not included in $d\mathbb{Z}$, then neither is $\mathrm{Pre}(d\mathbb{Z} \cap \mathrm{Ampl}^d(A, E))$. Let $x \in \mathrm{Pre}(d\mathbb{Z}) \setminus d\mathbb{Z}$. All counter values $x + a + e$ for $a \in A$ and $e \in E$ are included in the interval $\mathrm{Ampl}(A, E) + x$, a fortiori when e is chosen according to x and a such that $x + a + e \in d\mathbb{Z}$. In this case, $x \bmod d$ belongs to $\mathrm{Pre}(d\mathbb{Z} \cap \mathrm{Ampl}^d(A, E)) \setminus d\mathbb{Z}$, and it is outside $d\mathbb{Z}$ too.

Second, we prove the proposition by contrapositive: Suppose that $\mathrm{Pre}(d\mathbb{Z} \cap \mathrm{Ampl}^d(A, E))$ is included in $d\mathbb{Z}$. We just proved that it implies the inclusion of $\mathrm{Pre}(d\mathbb{Z})$ in $d\mathbb{Z}$. As a consequence, from any counter value outside $d\mathbb{Z}$, there exists a move of Adam such that for all moves of Eve, the next round begins outside $d\mathbb{Z}$ too, in particular it is impossible for Eve to have a winning strategy. Hence, d is in fact $\gcd(\mathrm{Win})$. □

We need to adapt this result because we do not know whether $Y \subseteq \mathrm{Win}$ and we look for a statement that allows us to find winning counter values. That is why we define the regularity interval $I_{(A,E)}(X)$ of a finite subset X of $\mathbb{Z}$ neither empty nor equal to $\{0\}$ in a robot game (A, E). The elements of this interval are X-reachable if, and only if, they are multiples of $\gcd(X)$.

- If $X \subset \mathbb{N}$, then $I_{(A,E)}(X) := (\tilde{F}(X) - \min(A) - \min(E) + d) + \mathrm{Ampl}^d(A, E)$, the lower bound of this interval is $\tilde{F}(X)$.
- If $X \subset -\mathbb{N}$, then $I_{(A,E)}(X) := (\tilde{F}(X) - \max(A) - \max(E) - d) + \mathrm{Ampl}^d(A, E)$, the upper bound of this interval is $\tilde{F}(X)$.
- Else, $I_{(A,E)}(X) := \mathrm{Ampl}^d(A, E)$.

Corollary 35: *Let Win be the winning set in a robot game (A, E), and $X \subset$ Win such that $\gcd(X) = d > \gcd(\mathrm{Win})$. Let $Y' = I_{(A,E)}(X) \cap d\mathbb{Z}$. Then we have $\mathrm{Pre}(Y') \setminus d\mathbb{Z} \neq \emptyset$. As a consequence, if Win $\not\subseteq d\mathbb{Z}$, then we can compute a certain element $d\mathbb{Z} \setminus$ Win in space polynomial*

in $|A|$ and $|E|$.

We apply this idea to the game in Figure 3.1 on page 57 and find that -2 is a winning counter value outside $-3\mathbb{N}$, because if Adam plays 3, then Eve can play -1 and win, and if Adam plays -1, then Eve can play 0, and in the next round she can play the difference between 3 and Adam's move to win. With the notations of the last proposition and of its corollary, we have $\tilde{F}(\{-3\}) = 0$, the interval $I_{(\{-1,3\},\{-1,0,4\})}(\{-3\})$ is $(0-3-4-3)+[\![-5,10]\!] = [\![-15,0]\!]$, and $\mathrm{Pre}(\{-15,-12,\ldots,0\})$ is not included in $3\mathbb{Z}$. We pick -2 in it. Since $\gcd(\{-2,-3\}) = 1$, we know that $\gcd(\mathrm{Win})$ is 1.

Theorem 36: *Algorithm 3 computes the winning set in a robot game in exponential time.*

Proof. (Termination) The only loop in the algorithm is in the first step. Each iteration either lowers the variable d', more precisely replaces it by one of its divisors, or assigns the variable d to the value of d', which makes the loop stop because this value is positive. The lowering of d' occurs less times than the greatest common divisor of $\mathrm{Pre}(\{0\}) \cup \{0\}$, and if this set is $\{0\}$, then the algorithm stops before the first step begins. □

Proof. (Correctness) Let Win be the actual winning set, and let $d = \gcd(\mathrm{Win})$.

- In the first step, the variable X is a subset of Win. We prove it by recurrence:
  - The step begins with $X = \mathrm{Pre}(\{0\}) \cup \{0\}$, which contains only winning counter values.
  - Let $X \subseteq \mathrm{Win}$, let $d' = \gcd(X)$. In the loop, when a counter value y is included in X, it belongs to $\mathrm{Pre}(I \cap d'\mathbb{Z})$, where I is the regularity interval of X. Thus Eve has a move to go from y to a subset of X-reachable counter values, which justifies that y is a winning counter value too.
- On the other hand, if no element of $\mathrm{Win} \backslash d\mathbb{Z}$ is found and included in X, then by Corollary 35, there exists none. It remains to look for elements of $\mathrm{Win} \backslash \langle X \rangle_{\mathbb{N}}$, necessarily in $d\mathbb{Z}$.
- We distinguish three cases to prove the second step.
  - If two counter values in X have opposite signs, then $\mathrm{Win} = \langle X \rangle_{\mathbb{N}} = d\mathbb{Z}$ by Lemma 29 on page 63.
  - Else if $X \subseteq \mathbb{N}$ and $-\mathbb{N} \cap \mathrm{Pre}(\mathrm{Ampl}(A,E) \cap d\mathbb{N}) \neq \emptyset$, then we also have $\mathrm{Win} = d\mathbb{Z}$. Actually we here prove this to be equivalent to the fact that two counter values in Win have opposite signs.

    ($\Leftarrow$) Let $x_0 \in -\mathbb{N} \cap \mathrm{Win}$. Consider a play π that starts at x_0 and in which Eve uses a winning strategy. The play π ends in 0 and every round finishes in winning counter values, i.e., multiples of d. Let $x \in -\mathbb{N}$ be the counter value in which a round in π ended and no more round ended in $-\mathbb{N}$ afterwards. Whatever Adam did, Eve forced the

Algorithm 3: Algorithm for solving robot games on the integer line.

Input: A robot game (A, E).
Result: A description of the winning set.
/* Require: Functions computing the sets that we use, as defined in the Section 3.1. */
begin
  $d \leftarrow 0$
  $X \leftarrow \mathrm{Pre}(\{0\}) \cup \{0\}$ /* to avoid handling gcd($\{0\}$) in the first step */
  if $X = \{0\}$ **then return** X

  /* Step 1. */
  while $d = 0$ **do**
    $d' \leftarrow \gcd(X)$
    $I \leftarrow I_{(A,E)}(X)$
    /* I is a set of X-reachable counter values with a large absolute value */
    $Y \leftarrow \mathrm{Pre}(I \cap d'\mathbb{Z})$
    /* From Y, Eve can force the next round to end at a counter value known to be winning */
    if $Y \setminus d'\mathbb{Z} \neq \emptyset$ **then** $X \leftarrow X \cup \{\min(Y \setminus d'\mathbb{Z})\}$
    /* minimum in absolute value */
    else $d \leftarrow d'$
    /* We know that d is gcd(Win): we exit the loop */

  /* Step 2. */
  if $X \not\subseteq \mathbb{N} \wedge X \not\subseteq -\mathbb{N}$ **then return** $d\mathbb{Z}$
  /* Lemma 29 */
  else
    $I \leftarrow \mathrm{Ampl}(A, E)$
    $b \leftarrow \tilde{F}(X)$
    if $X \subseteq \mathbb{N}$ **then**
      if $-\mathbb{N} \cap \mathit{Pre}(I \cap d\mathbb{N}) \neq \emptyset$ **then return** $d\mathbb{Z}$
      /* Lemma 29, second try */
      else $\mathit{Unbd} \leftarrow \{x \in d\mathbb{Z} \mid x > b\}$
    /* Half-line of winning counter values */
    else
      if $\mathbb{N} \cap \mathit{Pre}(I \cap -d\mathbb{N}) \neq \emptyset$ **then return** $d\mathbb{Z}$
      else $\mathit{Unbd} \leftarrow \{x \in d\mathbb{Z} \mid x < b\}$

    $G \leftarrow \mathrm{Restr}_d^b(A, E)$
    /* Between 0 and b, we compute the attractor on the restricted arena according to Proposition 27 */
  return $\mathit{Unbd} \cup \mathit{RestrAttr}(G)$

round that began in x to end in a nonnegative winning counter value. To sum up, x is a negative counter value in $\mathrm{Pre}(\mathrm{Ampl}(A,E) \cap d\mathbb{N})$.

Note that $\mathrm{Ampl}(A,E)$ necessarily contains negative counter values, else there would not be any positive winning counter value.

($\Rightarrow$) Let $x \in -\mathbb{N} \cap \mathrm{Pre}(\mathrm{Ampl}(A,E) \cap d\mathbb{N})$. In other words, for every move of Adam, Eve has a move such that a round that begins in x ends in a positive multiple of d in one round, and this multiple is less than $\max(A) + \max(E)$. Consider Eve's move as the image of Adam's move by a function $\varphi : A \to E$. If Eve plays the image by φ of the last move of Adam dk times, for $k \in \mathbb{N}$ big enough, then a great multiple of d, i.e., a counter value in $\langle X \rangle_{\mathbb{N}}$, is reached. This justifies that x is winning.

If $X \subseteq -\mathbb{N}$ and $\mathbb{N} \cap \mathrm{Pre}(\mathrm{Ampl}(A,E) \cap -d\mathbb{N}) \neq \emptyset$, we have the same result.

– Else, we know that all counter values in Win have the same sign. Suppose without loss of generality that $X \subseteq \mathbb{N}$. From a negative counter value, only negative or positive but surely losing counter values can be visited, therefore Win is included in $d\mathbb{N}$. We use Theorem 28 on page 62: every counter value above $\tilde{F}(X)$ is winning. Between 0 and $\tilde{F}(X)$, we decide the winner using the result of Proposition 27 on page 61 about the attractor on the restricted arena.

□

Proof. (Complexity) We consider the input size as $\sum_{w \in A \cup E} \log(|w|)$. The algorithm first computes the set $\mathrm{Pre}(\{0\})$, which contains at most $|E|$ counter values, all of them have a lower size than the input size.

Let us consider the loop in the first step of the algorithm. For any subset X of $\mathbb{Z}$, let d' be the greatest common divisor of X and let I be the regularity interval of X. The size of I is $2\ \gcd(X) + \min(A) + \max(A) + \min(E) + \max(E)$, one of its bounds is $\tilde{F}(X)$, and the size of a representation of this integer is at most twice the size of X. The size of the counter value y obtained in the loop using Pre on $I \cap d'\mathbb{Z}$ is bounded by a polynom in the size of the integers in X. There is a logarithmic number of iterations in the loop, because each assignment of d' sets it to one of its strict divisors.

We now look at the second step of the algorithm. It first checks whether two integers in X have opposite signs, and in case of fail it makes another test on the Pre of an interval included in the amplitude of the game. This can be done in polynomial time. If the second test fails too, then the bound $b := \tilde{F}(X)$ is computed, the arena $G := \mathrm{Restr}_d^b(A,E)$ is built and the reachability game on G is solved with the computation of an attractor, for a time complexity that is polynomial in the size of G. This size is linear in the value of $b + \max(A) - \min(A)$. With a binary encoding, the algorithm uses then exponential time. □

Let us illustrate the second step of the algorithm with the example in Figure 3.1 on page 57

again. We exit the first step with a subset $X = \{-2, -3\}$ of the winning set such that $\gcd(X) = 1$. Because $1 = -\min(E) < \max(A) = 3$, Adam wins from any positive counter value (Proposition 25 on page 59), it is indeed impossible that $\mathrm{Pre}(\{-2, -1, 0\})$ contains any positive counter value. Every nonpositive $\{-2, -3\}$-reachable counter value, i.e., every nonpositive counter value but -1, is winning. We only have to decide whether Eve wins from -1, and it is not the case because Adam can play 3 every time, which guarantees that only positive counter values are visited after the first move. The algorithm decides it when it calls RestrAttr on the arena $\mathrm{Restr}_1^{-1}(A, E)$.

3.2 Lower complexity bound for dimension one

We are now showing EXPTIME-hardness of robot games. In order to do this, we give the definition of countdown games, presented in [JLS07], which are games with one positive and strictly decreasing counter. We then introduce a variant of countdown games and show two successive reductions, from countdown games to our variant and then from this variant to robot games.

A *countdown game* between two players 1 and 2 is represented by a pair $((S, T), c_0)$ where S is a finite set of *locations*, $T \subseteq S \times (\mathbb{N} \setminus \{0\}) \times S$ is a set of weighted transitions and $c_0 \in \mathbb{N} \setminus \{0\}$. We consider that S has a particular location s_0. Configurations in the game are pairs $(s, c) \in S \times \mathbb{N}$, where c is a counter value. A play is a sequence of moves, done in the following way: From a configuration (s, c), initially (s_0, c_0), player 1 chooses a value $d \leq c$ called *duration* such that there exists a transition in T with s as first component and d as second component, then player 2 chooses (s, d, s') among these transitions. This move updates the configuration to $(s', c - d)$.

The winner of a play in a countdown game is determined once the play is blocked. Because only nonnegative integers appear in the configurations and positive integers in the transitions, the game is finite and ends when player 1 cannot find any duration to make a move. At this point, player 1 wins if, and only if, the counter value is 0. Determining the winner in countdown games is EXPTIME-complete, according to [JLS07].

We now define *restricted countdown games*: On the one hand, the winning condition for player 1 is now that the play ends in $(\bot, 0)$ for a particular sink $\bot \in S$, i.e., there are no transitions with $\bot$ as first component; on the other hand, if there are two transitions (s_1, d_1, s_1') and (s_2, d_2, s_2') in T with $d_1 = d_2$, then $s_1 = s_2$; in other words, a duration is specific to a starting location.

Proposition 37: *Countdown games reduce in polynomial time to restricted countdown games.*

Proof. There are two steps in the construction. Consider an arbitrary countdown game $G = ((S, T), c)$, where $S = \{s_0, \ldots, s_{n-1}\}$. Let d' be the least positive integer that does not appear in any transition in T. First, we build the countdown game $G' = ((S \cup \{\bot\}), T \cup \{(s, d', \bot) \mid s \in S\}, c + d')$, The winning condition for player 1 in G' is to reach $(\bot, 0)$.

Note that player 1 wins a play π in G' if, and only if, in the last move of π, the configuration is (s, d') for any location s, and player 1 chooses d', in order that player 2 can only pick the transition

$(s, d', \bot)$. The partial play from $(s_0, c + d')$ to (s, d'), corresponds in G to a play that starts at (s_0, c) and ends in $(s, 0)$, where player 1 wins.

Second, we build from G' a restricted countdown game that we prove to be equivalent to G. Let $G'' = ((S \cup S' \cup \{\bot\}, T''), 2N(c+d'))$, where $S' = \{s'_1, \ldots, s'_{N-1}\}$ and $T'' = \{(s_0, 2nd, s) \mid (s_0, d, s) \in T'\} \cup \{(s_i, i, s'_i), (s'_i, 2nd - i, s_j) \mid (s_i, d, s_j) \in T'\}$. Matching transitions are $(s_0, 2nd, s) \in T''$ and $(s_0, d, s) \in T'$, as well as $(s'_i, 2nd - i, s_j) \in T''$ and $(s_i, d, s_j) \in T'$. Matching plays are $\pi'' \in T''$ and $\pi' \in T'$ such that, when we exclude the moves from $s \in S$ to s' in π'', the transitions of every move in π'' and in π' match.

The game G'' is a restricted countdown game because the duration of a transition that starts in $s_i \in S$ is a multiple of $2N$ plus i and the duration of a transition that starts in $s'_i \in S'$ is a multiple of $2N$ minus i.

Moreover, player 1 wins in G'' if, and only if, he wins in G', thus in G. Indeed, consider matching plays π'' of G'' and π' of G'. When the location is in $S \cup \{\bot\}$, the counter value in π'' is $2n$ times the counter value in π', because at the beginning of π'' the configuration is $(s_0, 2n(c_0 + d'))$ and at the beginning of π' the configuration is $(s_0, c_0 + d')$.

Hence, a play in G'' reaches $(\bot, 0)$ if, and only if, the matching play in G' reaches $(\bot, 0)$. □

Theorem 38: *Given a robot game on $\mathbb{Z}$ and an initial counter value, determining whether Eve has a winning strategy from this counter value is EXPTIME-hard.*

We prove the theorem by a reduction from restricted countdown games to unidimensional robot games. Let $((S, T), c)$ be a restricted countdown game, where we suppose without loss of generality that $S = [\![0, n-1]\!]$, $s_0 = 0$ and $\bot = n - 1$. We denote by $D = \{d_0, \ldots, d_{h-1}\}$ the set of values that appear in T as durations. Let $k = \lfloor \log_4(c) \rfloor + 1$ and $k' = \lfloor \log_4(k) \rfloor + 1$.

We write the counter value in the robot game in base 4 and with $h + n + k + k' + 1$ digits, which we split in four parts. These parts encode *durations*, *locations*, *value* in the countdown game and *controls*, from the least significant digit to the most significant one. We explain these notions after the presentation of the sets of moves U and V. The initial counter value is $4^h + c_0 \cdot 4^{h+n} + k \cdot 4^{h+n+k} + 4^{h+n+k+k'}$, it corresponds to no duration given, the initial location, the value c_0 in the countdown game, and a default value for the control part.

Let us use an example to see how we represent a configuration in a restricted countdown game by the counter value in the robot game. Consider the countdown game pictured in Figure 3.2. We represent in Figure 3.3 the first move of a play that starts at 0 with value 8 (first line) and where player 1 chooses duration 3 (second line), after which player 2 moves to the location 1 (third line). We set $D = \{1, 2, 3, 4, 5, 6\}$.

For simplicity, we decide that Eve begins in the robot game, but the winning condition is still that the counter value becomes 0 after a turn of Eve. It remains computationally equivalent.

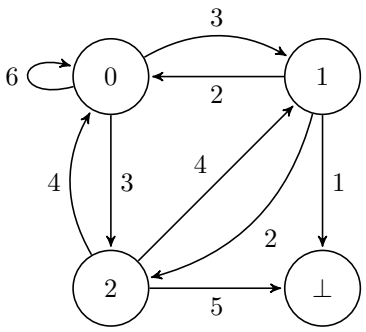

Figure 3.2 – Restricted countdown game.

duration	location	countdown	control
0 0 0 0 0 0	1 0 0 0	0 2	2 1
0 0 1 0 0 0	0 0 0 0	1 1	2 1
0 0 0 0 0 0	0 1 0 0	1 1	2 1

Figure 3.3 – Counter values in the robot game.

We first give the moves of both players in the robot games as codes, in order to explain the way we encode a play in the restricted countdown game. Intuitively, because we split the counter in four parts, a risk appears that the encoding no longer corresponds to a configuration in the reduced countdown game, for example because of a carry. We prove that if a player tries to create such a bad behaviour, then the other one can react with a winning strategy.

In a restricted countdown game, let us write s_d for the location uniquely determined by a duration d.

The codes for the set of moves of Adam are $\{\textsc{Duration}\ d\ \textsc{Goto}\ s' \mid (s_d, d, s') \in T\}$, which correspond to the choice by player 2 of the next location according to the given duration.

The codes for the set of moves of Eve are $\{\textsc{State}\ s\ \textsc{Choose}\ d \mid \exists s', (s, d, s') \in T\}$, which correspond to the choice by player 1 of an available duration; $\{\textsc{Finish}\}$, played when the winning configuration is reached; $\{\textsc{Cancel}\ (d, s')\ \textsc{Erase}\ (j, a) \mid (s_d, d, s') \in T,\ 0 \leq j < k,\ 0 \leq a \leq 3\}$, to modify the third and fourth part of the counter value and eventually reach 0 after a deviating move of Adam, a notion that we introduce in the following; and $\{\textsc{Cancel}\ (d, s')\ \textsc{Remove}\ d' \mid (s_d, d, s') \in T,\ d \neq d'\}$, to point out a deviating move of Adam, i.e., cancel the effect of the last move and subtract the real duration that was chosen. When we do not specify the parameters like s and d in the codes, we write the type of the moves, for example State/Choose.

We call *good encoding* a sequence that alternates moves of Eve and moves of Adam such that:

- The first move is $\textsc{State}\ s_0\ \textsc{Choose}\ d$ for a certain d.
- The last move is Finish and the move before is $\textsc{Duration}\ d\ \textsc{Goto}\ \bot$ for a certain d.
- There are neither Cancel/Erase nor Cancel/Remove nor other Finish moves.

- For two consecutive moves STATE s CHOOSE d and DURATION d' GOTO s', we have $d = d'$.
- For two consecutive moves DURATION d GOTO s and STATE s' CHOOSE d', we have $s = s'$.

All sequences that are not the prefix of a good encoding and that have no good encoding as a prefix are *bad encodings*, and the first move that refutes in this case the first or one of the three last properties of a good encoding is called *deviating move*. A sequence that only refutes the second property is neither a good nor a bad encoding, and we deal separately with sequences that continue after a finish move.

Note that for each play in the restricted countdown game the players have the possibility to build with their moves in the robot game a good encoding and the codes of this good encoding trace the play. The hard part is to handle bad encodings. To understand how it can be done, let us give the integers that correspond to each code.

- DURATION d_i GOTO $s' == -4^i + 4^{h+s'}$;
- STATE s CHOOSE $d_i == 4^i - 4^{h+s} - d_i \cdot 4^{h+n}$;
- FINISH $== -4^{h+n-1} - k \cdot 4^{h+n+k} - 4^{h+n+k+k'}$;
- CANCEL (d_i, s') ERASE (j, a) $==$ $-$(DURATION d_i GOTO s')$-a \cdot 4^{h+n+j} - 4^{h+n+k}$;
- CANCEL (d_i, s') REMOVE d_j $==$ $-$(DURATION d_i GOTO s')$-4^j - 4^{h+n+k+k'}$.

It appears that every move of Adam is positive and every move of Eve is negative. Moreover, any move of Adam move plus any move of Eve is negative, hence, if the counter value becomes negative, then Adam wins.

The next proposition shows the need for both players to build good encodings.

Proposition 39: *If a sequence is a bad encoding, then the adversary of the player who has played the deviating move has a winning strategy from this move onwards.*

Proof. Let us consider a bad encoding and every possibility for the deviating move:

- A move of Adam DURATION d_i GOTO s' whereas the expected duration was d_j.

  In this case, the counter value has the i^{th} digit at 3. Eve then has the occasion to play CANCEL (d_i, s') REMOVE d_j, in order that the first two parts of the counter value become 0. From this point onwards, Eve can cancel the effect of every further move of Adam and erase step by step every digit of the third part of the counter until its value becomes 0.

  This case is illustrated in the Figure 3.4. The first line corresponds to the second line in the Figure 3.3. Imagine that Adam plays the deviating move DURATION 6 GOTO 0 (second line).

Eve can react with Cancel (6,0) Remove 3 (third line), and then, whatever Adam does (fourth and sixth line), Eve cancels the effect of every move and erases the first and second digits of the third part (fifth and seventh line), which makes her win because the counter value is 0.

- A move of Eve State s Choose d_i whereas the expected location was s'.

  In this case, the counter value has the $h + s^{\text{th}}$ digit at 3. Now, Adam can take advantage of this error and play a move with Goto s when, and only when, the $h + s^{\text{th}}$ digit has been lowered to 2 by the previous move of Eve, therefore this digit will always be 3 after a move of Adam and never 0 again after a move of Eve, because none of her moves permits to increase a digit of the second part or to decrease it by 2 or more, even with carries. In particular, the counter value cannot become 0.

- A Cancel/Remove move of Eve.

  Here, the first part of the counter value had only digits at 0 just before because Adam did not do a deviating move and now one digit is at 3, let us say it is the i^{th} one. Adam will always use moves with Duration d_i when the i^{th} digit is 3 and other moves when it is 2. Hence, the i^{th} digit can no longer be put to 0 after Eve plays, therefore Adam wins.

- A Cancel/Erase move of Eve.

  Here, the fourth part of the counter value has been reduced, hence the move Finish, which would lead to a negative counter value, should be avoided by Eve. In other words, Adam just has to match the duration of his move to the one that Eve chose right before, like in a good encoding, to be sure that he wins. Indeed, the only possibility for Eve to win is now to use a Cancel/Remove move, but she will lose whenever she does this meanwhile the first part of the counter value has only digits at 0.

□

0 0 1 0 0 0	0 0 0 0	1 1	2 1	Expected duration: 3
0 0 1 0 0 3	0 0 0 0	1 1	2 1	Duration 6 Goto 0
0 0 0 0 0 0	0 0 0 0	1 1	2 0	Cancel (6,0) Remove 3
0 0 0 0 3 3	3 3 3 0	1 1	2 0	Duration 5 Goto $\perp$
0 0 0 0 0 0	0 0 0 0	0 1	1 0	Cancel $(5, \perp)$ Erase $(1, 1)$
0 0 0 0 3 3	3 3 3 0	0 1	1 0	Duration 5 Goto $\perp$
0 0 0 0 0 0	0 0 0 0	0 0	0 0	Cancel $(5, \perp)$ Erase $(2, 1)$

Figure 3.4 – Deviating move of Adam and reaction of Eve.

We now restrict to good encodings, for which the next proposition decides the winner depending on the winner of the corresponding play in the countdown game. We first need the following lemma.

Lemma 40: *Consider a prefix of a good encoding without any* FINISH *move. The following invariants hold:*

- *The digits in the first part of the counter value are all* 0 *after a move of Adam and all* 0 *except one* 1 *after a move of Eve.*
- *The digits in the second part of the counter value are all* 0 *after a move of Eve and all* 0 *except one* 1 *after a move of Adam.*

Proof. At the beginning of the play, before the first move of Eve, there is one 1 and other digits are 0 in the second part, and all the digits in the first part are 0. This can also be seen in the Figure 3.3. Consider a good encoding. The following alternation happens: STATE/CHOOSE moves of Eve erase the 1 in the second part and increment a digit, hence a 0, in the first part; and moves of Adam erase the 1 that appeared in the first part and increment a 0 in the second part. □

Proposition 41: *Consider a good encoding π' built from a play π in the restricted countdown game. If player* 1 *wins π, then Eve wins any play that begins with the prefix π' in the robot game when he plays the* FINISH *move, else Adam has a winning strategy after the* FINISH *move.*

Proof. We look at the evolution of the counter value along a good encoding. Every DURATION d cancels the effect of the previous CHOOSE d, every STATE s cancels the effect of the previous GOTO s, the move STATE s_0 zeroes the 1 in the initial counter value, and FINISH zeroes the digit that corresponds to $\perp$ and the fourth part. In other words, all parts except possibly the third one are zero at the end of a good encoding. Here, we do not consider possible carries from the third to the fourth part, which make Eve lose. As for the third part, at the beginning, it represents the initial value in the countdown game and Eve subtracts from it the values of the durations chosen by player 1 in the simulated play. The counter value is also 0 at the end of the good encoding in the robot game if, and only if, the corresponding play in the countdown game ends in the configuration $(\perp, 0)$.

Let us show why we need the FINISH move. According to Lemma 40, Eve cannot win if she plays STATE/CHOOSE moves forever, even if she tries to make a carry appear from the third to the fourth part of the counter value, because there will always be a digit at 1 in the first part of the counter. However, with a FINISH move, no digit is incremented in the first part of the counter. Hence, Eve should use this move at least once at the end of a good encoding.

Note that, in particular, Eve loses if she subtracts from the third part more than the initial value in the countdown game.

Now, we present the winning strategy for Adam if Eve did not win at the moment where she played FINISH. The fourth part of the counter is now nullified, hence Eve can afterwards only use STATE/CHOOSE moves because other moves would make the counter value negative. Consequently, Adam can do a move with GOTO $\perp$ and guarantee at the next step that the digit that corresponds to $\perp$ is never 0 again. □

We conclude from Propositions 39 and 41 that Eve has a winning strategy in the robot game if, and only if, player 1 has a winning strategy in the countdown game. Indeed, both players need to generate a good encoding, else they know that they will lose, thus, it is just a matter of checking whether Eve can enforce the good encoding to make her win.

To sum up, the algorithm that we give in the previous section is asymptotically optimal, and the problem of determining whether a given integer is winning in a robot game of dimension one is EXPTIME-complete. Moreover, for this dimension, building the whole winning set and giving a description of it is also doable in exponential time.

3.3 Undecidability for dimension three

In this section, we prove the undecidability of the problem of determining the winner of a robot game in dimension three, by a reduction of a variant of the halting problem of Minsky machines. Our proof is in three steps. In each step, we introduce a new argument to prove the undecidability of the problem of determining the winner in an increasingly stronger model of robot games.

For the first reduction, we extend the definition of robot games by giving control states to Eve. The term *control states* of Eve means that we represent states of a Minsky machine so that, unlike in counter reachability games, Adam has no influence on transitions from one control state of Eve to another.

For the second reduction, we show how to encode Eve's control states in several additional counters. The number of counters depends linearly on the number of control states of Eve, thus on the size of the original Minsky machine.

For the third reduction, we keep two counters inherited from the Minsky machine and we introduce a third counter that stores all information of the additional counters from the second reduction.

3.3.1 Minsky machines

Before giving the first reduction, we present Minsky machines, introduced in [Min67], a crucial model for our proof. A *deterministic two-counter Minsky machine* (2CM) is a pair (Q, T), where Q is a finite set of states and $T \subseteq Q \times \{c_1{+}{+}, c_1{-}{-}, c_1 == 0, c_2{+}{+}, c_2{-}{-}, c_2 == 0\} \times Q$ is a finite set of labelled transitions to increment, decrement or test to zero one of the counters. In a deterministic two-counter Minsky machine, the set Q contains an initial state q_0 and a sink state $\perp$, such that there is no outgoing transition from $\perp$. Moreover, from all $q \in Q \setminus \{\perp\}$, either there is only one outgoing transition with 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 $(q, (y, z)) \in Q \times \mathbb{N}^2$, 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 $(q_0, (0, 0))$ and follows transitions of the machine, insofar as for

two consecutive configurations $(q,(y,z))$ and $(q',(y',z'))$, one of the following holds:

- $y' = y+1, z' = z$ and $(q, c_1\texttt{++}, q') \in T$, the first counter is incremented upon the transition;
- $0 \leq y' = y-1, z' = z$ and $(q, c_1\texttt{--}, q') \in T$, the first counter must be positive and it is decremented upon the transition;
- $y' = y = 0, z' = z$ and $(q, c_1 \texttt{==} 0, q') \in T$, the first counter must be zero before taking the transition;
- $y' = y, z' = z+1$ and $(q, c_2\texttt{++}, q') \in T$, the second counter is incremented upon the transition;
- $y' = y, 0 \leq z' = z-1$ and $(q, c_2\texttt{--}, q') \in T$, the second counter must be positive and it is decremented upon the transition;
- $y' = y, z' = z = 0$ and $(q, c_2 \texttt{==} 0, q') \in T$, the first counter must be zero before taking the transition;

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 to zero.

The *halting problem* of 2CM is to decide, given a 2CM, whether the run reaches a configuration with state $\perp$, in other words whether the run halts.

Theorem 42 ([Min67])**:** *The halting problem of 2CM is undecidable.*

Reductions from the halting problem of 2CM can be seen in the literature to prove undecidability of games on VASS, with Minsky machines that are deterministic, like in [BJK10], or nondeterministic, like in [ABD08].

Because Eve's objective in a robot game is to reach a counter vector, we introduce another decision problem for Minsky machines, and we denote this new decision problem by Reach-2CM: given a 2CM (Q,T), decide whether the run ρ of (Q,T) returns at least once to a configuration in $Q \times \{(0,0)\}$. Note that the sink $\perp$ of the 2CM is no longer particular when we consider the problem Reach-2CM.

Proposition 43: *Reach-2CM is undecidable.*

Proof. We reduce the halting problem of 2CM to Reach-2CM.

Let (Q,T) be a deterministic two-counter Minsky machine with initial state q_0. We build a deterministic two-counter Minsky machine (Q',T') such that the run in (Q,T) from $(q_0,(0,0))$ reaches a configuration in $\{\perp\} \times \mathbb{N}^2$ if, and only if, the run in (Q',T') from a configuration $(q_0',(0,0))$ returns to a configuration in $Q \times \{(0,0)\}$.

To do this, we first ensure that whenever the configuration is $(q,(0,0))$ in (Q',T') after the first step of the run, then $q=\bot$. Intuitively, the run ρ' on (Q',T') simulates the run ρ on (Q,T) with the feature that the value of each counter in each step of ρ' is either equal to the value of the same counter in the corresponding step of ρ, or equal to one plus the value of the same counter. In the reduction, we call *shift* the fact that the value of a counter in a step of ρ' is equal to one plus the value of the same counter in the corresponding step of ρ. We ensure that, at each step until $\bot$ is reached, there is at least one shift, and the run starts with a shift of both counters upon transitions from q_0' to q_0'' and from q_0'' to q_0, where q_0' is a fresh state, fixed to be the initial state of (Q',T'), and q_0'' is also fresh. Before any transition that decrements a counter or tests it to zero, there is a shift on both counters, and the shift is cancelled for the counter that the transition modifies or tests to zero. This is done with a gadget of three transitions, and we introduce fresh states, which we gather in a set that we denote by Q_2, for such gadgets : f_q, s_q and s_q', for $q\in Q$ with two outgoing transitions. If the sink $\bot$ is reached in (Q,T), then all counters are decreased to zero in (Q',T'), which is done in two fresh states $\bot'$ and $\bot''$, hence the instance of Reach-2CM is positive.

Formally, let $Q'=Q\cup Q_2\cup\{q_0',q_0'',\bot',\bot''\}$. The set Q_2 contains all states f_q, s_q and s_q', for $q\in Q$, which are the **f**irst and **s**econd intermediary states to cancel the shift of a counter, apply the effect of an outgoing transition, and put back the shift.

The set T' gathers all gadgets that replace transitions in T, as we show in Figure 3.5. Assume that there are two outgoing transitions from q, so a decrement of a counter, say the first counter, and a zero test of the same counter. There are five transitions in T' that replace the two outgoing transitions, which we denote by $(q,c_1\texttt{--},r)$ and $(q,c_1\texttt{==}0,s)$, from q in T: one from q to f_q that decrements the first counter, one from f_q to s_q that decrements the first counter, one from f_q to s_q' that tests the first counter to zero, one from s_q to r that increments the first counter, and one from s_q' to s that increments the first counter. Otherwise, there is one outgoing transition from q, with an increment. All transitions of T with an increment are also in T' without any change. At the beginning of the run, to initiate the shift of the counters, T' contains two transitions $(q_0',c_1\texttt{++},q_0'')$ and $(q_0'',c_2\texttt{++},q_0)$. Moreover, T' contains additional transitions $(\bot,c_1\texttt{--},\bot)$, $(\bot,c_1\texttt{==}0,\bot')$, $(\bot',c_2\texttt{--},\bot')$ and $(\bot',c_2\texttt{==}0,\bot'')$. Hence, when the run reaches $\bot$ in (Q',T'), the counters are emptied and $\bot''$ is reached.

Formally, the set T' is the union of the following sets of transitions :

- $\{(q,c_1\texttt{++},r)\mid(q,c_1\texttt{++},r)\in T\}\cup\{(q,c_2\texttt{++},r)\mid(q,c_2\texttt{++},r)\in T\}$;
- $\{(q,c_1\texttt{--},f_q),(f_q,c_1\texttt{--},s_q),(s_q,c_1\texttt{++},r),(f_q,c_1\texttt{==}0,s_q'),(s_q',c_1\texttt{++},s)$, where $(q,c_1\texttt{--},r),(q,c_1\texttt{==}0,s)\in T\}$ (see Figure 3.5);
- $\{(q,c_2\texttt{--},f_q),(f_q,c_2\texttt{--},s_q),(s_q,c_2\texttt{++},r),(f_q,c_2\texttt{==}0,s_q'),(s_q',c_2\texttt{++},s)$, where $(q,c_2\texttt{--},r),(q,c_2\texttt{==}0,s)\in T\}$;
- $\{(\bot,c_1\texttt{--},\bot),(\bot,c_1\texttt{==}0,\bot'),(\bot',c_2\texttt{--},\bot'),(\bot',c_2\texttt{==}0,\bot'')\}$.

Note that there are missing transitions with a zero test in (Q',T') from the states q where we have created a transition to f_q. This is no problem since the transition to f_q is always the one that

can be executed because there is a shift on both counters when original states of (Q, T) are visited in (Q', T'). Still, it would be possible to add another state that gathers all the missing transitions and that has one outgoing transition to itself with label c_1++, but the construction and above all the drawings would become heavier. Anyway, we can see (Q', T') as a deterministic machine.

After the two initial transitions in (Q', T'), the runs proceed in parallel in (Q', T') and in (Q, T), except that the shift is cancelled for one counter while crossing gadgets, each of which matches one transition in (Q, T). As a consequence, when the state is not $\perp$, at least one counter is positive because of the shift.

We deduce that it is possible to reach a configuration with both counters at zero in (Q', T') from $(q'_0, (0, 0))$ if, and only if, a configuration with state $\perp$ is reachable in (Q, T) from $(q_0, (0, 0))$. □

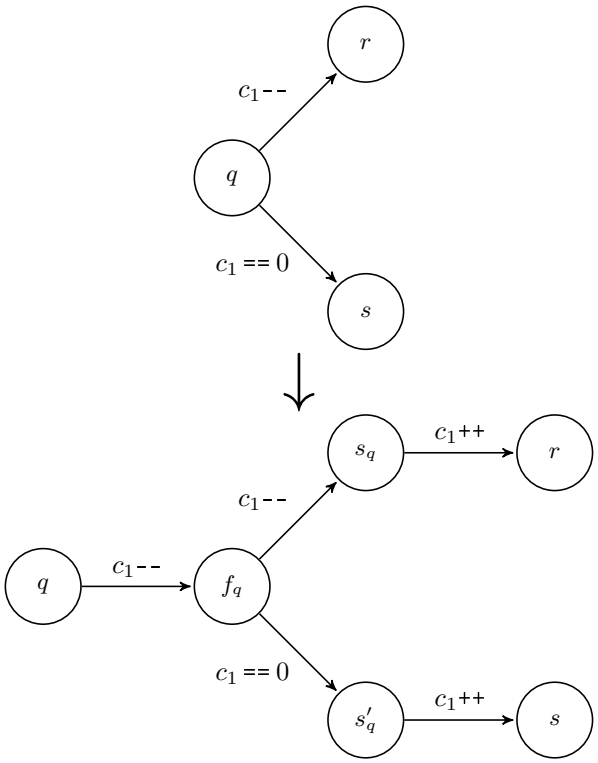

Figure 3.5 – Gadget to replace transitions $(q, c_1\texttt{--}, r)$ and $(q, c_1 == 0, s)$ in the reduction from the halting problem of 2CM to Reach-2CM.

Let us give an example to illustrate the reduction. Figure 3.6 represents a Minsky machine, in which the sink is reachable from the state q_0 after taking the loop once. The Minsky machine that we obtain with our reduction is represented in Figure 3.7, and we show corresponding runs in Figure 3.8.

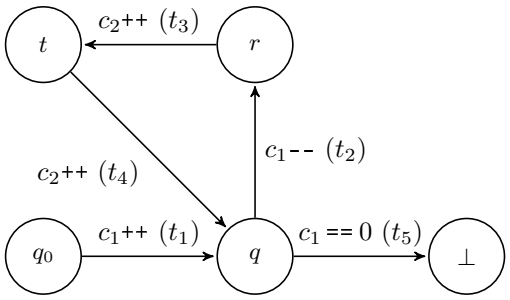

Figure 3.6 – Minsky Machine (Q, T).

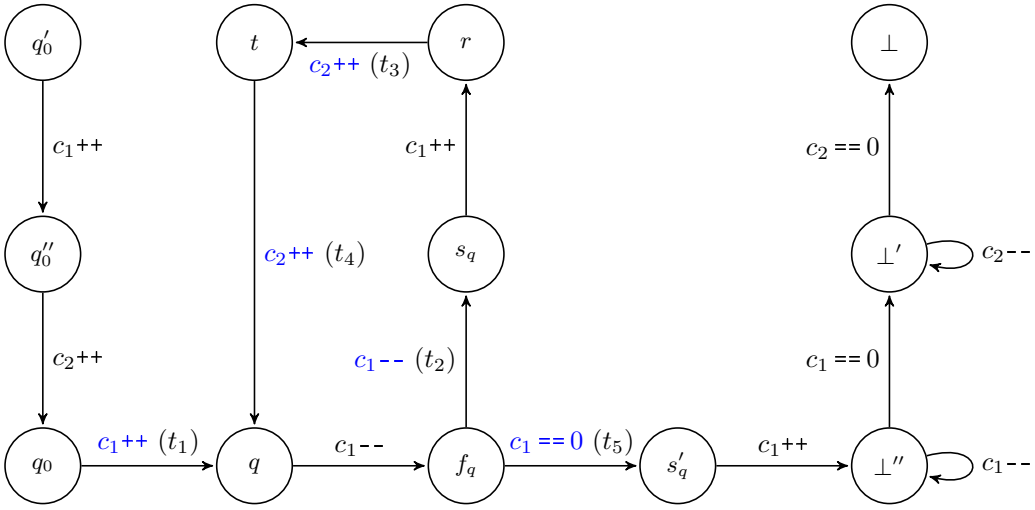

Figure 3.7 – Minsky Machine (Q', T') obtained from (Q, T) by the reduction in the proof of Proposition 43. Original transitions of (Q, T) are in blue.

Transition in (Q,T)	Configuration in (Q,T)	Configuration in (Q',T')	Transition in (Q',T')
Initial configuration	$(q_0,(0,0))$	$(q'_0,(0,0))$	Initial configuration
	–	$(q''_0,(1,0))$	$(q'_0,c_1{+}{+},q''_0)$
	–	$(q_0,(1,1))$	$(q''_0,c_2{+}{+},q_0)$
$(q_0,c_1{+}{+},q)$	$(q,(1,0))$	$(q,(2,1))$	$(q_0,c_1{+}{+},q)$
	–	$(f_q,(1,1))$	$(q,c_1{-}{-},f_q)$
$(q,c_1{-}{-},r)$	$(r,(0,0))$	$(s_q,(0,1))$	$(f_q,c_1{-}{-},s_q)$
	–	$(r,(1,1))$	$(s_q,c_1{+}{+},r)$
$(r,c_2{+}{+},t)$	$(t,(0,1))$	$(t,(1,2))$	$(r,c_2{+}{+},t)$
$(t,c_2{+}{+},q)$	$(q,(0,2))$	$(q,(1,3))$	$(t,c_2{+}{+},q)$
	–	$(f_q,(0,3))$	$(q,c_1{-}{-},f_q)$
$(q,c_1 == 0,\bot)$	$(\bot,(0,2))$	$(s'_q,(0,3))$	$(f_q,c_1 == 0,s'_q)$
Accepting run	(sink)	$(\bot'',(1,3))$	$(s'_q,c_1{+}{+},\bot'')$
	–	$(\bot'',(0,3))$	$(\bot'',c_1{-}{-},\bot'')$
	–	$(\bot'',(0,3))$	$(\bot'',c_1 == 0,\bot')$
	–	$(\bot',(0,2))$	$(\bot',c_2{-}{-},\bot')$
	–	$(\bot',(0,1))$	$(\bot',c_2{-}{-},\bot')$
	–	$(\bot',(0,0))$	$(\bot',c_2{-}{-},\bot')$
	–	$(\bot,(0,0))$	$(\bot',c_2 == 0,\bot)$
	–	(sink)	Accepting run

Figure 3.8 – Corresponding runs of (Q,T) and (Q',T').

3.3.2 First step: Undecidability of robot games with states in dimension two

In a first step towards the proof of undecidability of robot games in dimension three, we consider an extension of robot games: *robot game with states*. In our extension, Eve has control states, depending on which her move set changes. We prove that the problem of determining the winner of a robot game with states is undecidable in dimension two, by a reduction from Reach-2CM.

A robot game with states in dimension two with a finite set Q of control states is a pair (A,E), where A is a finite subset of $\mathbb{Z}^2$ and E is a finite subset of $Q\times\mathbb{Z}^2\times Q$. Robot games with states are played like robot games, except that a configuration is a pair (q,v) consisting of Eve's control state q and a counter vector $v\in\mathbb{Z}^2$. Eve updates her control state when she makes a move: in (q,v) for any vector v, only moves of the form (q,x,r) are enabled, and with one such move the new configuration is $(r,v+x)$. Eve wins if, and only if, both counters are zero after her turn, no matter which is her control state. The decision problem associated with robot games with states asks whether Eve has a winning strategy from a given configuration. We call this problem *2RGS* in dimension two.

Theorem 44: *Reach-2CM reduces to 2RGS.*

Proof. Let (Q,T) be a 2CM with initial state q_0. Let us explain how to build Eve's control states

in a robot game with states such that Eve has a winning strategy if, and only if, the run of (Q, T) returns to a configuration in $(0, 0)$. Most of Eve's moves in the robot game with states simulate transitions of the 2CM, and there is additional information about the positivity of the 2CM counters. This additional information is stored for each counter as a flag b or $B \in \{0, +\}$, which we include in Eve's control state.

Let us introduce a function $\delta : \mathbb{N} \to \{0, +\}$ such that $\delta(0) = 0$ and $\delta(n) = +$ for all $n > 0$, There is a *matching* between the counter value y and the flag b if, and only if, $b = \delta(y)$. We also say that the counter value y and the flag b *match*. In our reduction, the values of both counters will match the corresponding flag all along a play in the robot game with states that we build.

Let $Q' = (Q \cup \{\top\}) \times \{0, +\}^2$, where $\top$ is a sink such that $\top \notin Q$. Eve's set of moves is built so that the set $\{\top\} \times \{0, +\}^2$ cannot be exited. To have lighter notations, a control state $(q, (b, B)) \in Q'$ is denoted by q_{bB}. We construct a robot game with states (A, E), where $A = \{(0, 0), (1, 0)\}$ and $E \subseteq Q' \times \mathbb{Z}^2 \times Q'$ contains moves of two types.

First type, Eve's *regular moves*:

- For each transition $(r, c_1\texttt{++}, s) \in T$: four moves $(r_{bB}, (4, 0), s_{+B})$, for $b, B \in \{0, +\}$;
- For each transition $(r, c_1\texttt{--}, s) \in T$: four moves $(r_{+B}, (-4, 0), s_{bB})$, for $b, B \in \{0, +\}$;
- For each transition $(r, c_1 \texttt{==} 0, s) \in T$: two moves $(r_{0B}, (0, 0), s_{0B})$, for $B \in \{0, +\}$;
- For each transition $(r, c_2\texttt{++}, s) \in T$: four moves $(r_{bB}, (0, 1), s_{b+})$, for $b, B \in \{0, +\}$;
- For each transition $(r, c_2\texttt{--}, s) \in T$: four moves $(r_{b+}, (0, -1), s_{bB})$, for $b, B \in \{0, +\}$;
- For each transition $(r, c_2 \texttt{==} 0, s) \in T$: two moves $(r_{b0}, (0, 0), s_{b0})$, for $b \in \{0, +\}$.

With these moves, Eve simulates the run of the 2CM. Note that she can maintain the matching between the counter values and the flags: for example, if a move decrements the second counter from 1 to 0, then Eve updates B from $+$ to 0. We explain later the factor 4 in the updates of the first counter.

Second type, Eve's *emptying moves*:

- For each $q \in Q$: four moves $(q_{bB}, (-1, 0), \top_{bB})$, for $b, B \in \{0, +\}$;
- $\{(\top_{++}, (-4 - e, -1), \top_{bB}) \mid e \in \{0, 1\}, b, B \in \{0, +\}\}$;
- $\{(\top_{0+}, (-e, -1), \top_{0B}) \mid e \in \{0, 1\}, B \in \{0, +\}\}$;
- $\{(\top_{+0}, (-4 - e, 0), \top_{b0}) \mid e \in \{0, 1\}, b \in \{0, +\}\}$;
- $\{(\top_{00}, (-e, 0), \top_{00}) \mid e \in \{0, 1\}\}$.

With these moves, Eve's control state enters $\{\top\} \times \{0, +\}$ and stays there until the end of the play. Eve goes to $\{\top\} \times \{0, +\}^2$ and decrements the first counter with the first emptying move, and afterwards she may or may not decrement the first counter with the last four emptying moves, while she also decreases the first (respectively second) counter if, and only if, the first (respectively second) flag is $+$. When Eve's control state is in $\{\top\} \times \{0, +\}^2$, the control state may change, but only by changing a $+$ flag, or both, to a 0 flag.

To avoid that Eve wins trivially every play in the robot game with states, we do not use $((q'_{00}), (0, 0))$ as 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 $(q, (y, z))$ and we define $\bar{b} = \delta(y)$ and $\bar{B} = \delta(z)$. The initial configuration in the robot game with states is then $(q_{\bar{b}\bar{B}}, (4y, z))$.

We now prove the correctness of the reduction: Eve has a winning strategy if, and only if, the instance of Reach-2CM is positive, in other words if, and only if, the run of the 2CM returns to a configuration where both counters are zero. First, we prove that, if the instance of Reach-2CM is positive, then Eve has a winning strategy based on a simulation of the run of the 2CM. Second, we prove that Eve needs to ensure that the first counter is a multiple of 4 after each of her turns, otherwise she loses. Finally, we prove that, if the instance of Reach-2CM is negative, then Eve has no winning strategy, and our proof is based on the following properties: Eve needs to ensure that the flags of her control state and the counter values match, and the first counter must be a multiple of 4 after all turns of Eve.

We begin with the proof that Eve has a winning strategy if the instance of Reach-2CM is positive. Suppose that the run ρ of the 2CM returns to a configuration where both counters are zero. Eve's strategy, which we present in the next two paragraphs, is based on ρ, and depends on the moves that Adam uses during the play.

As long as Adam plays $(0, 0)$, Eve's strategy is to simulate ρ using her regular moves. She updates at each step her control state to q_{bB}, where q is the current state of the 2CM in ρ and b, B match the counter values.

As soon as Adam plays $(1, 0)$, the simulation of the run of the 2CM stops and Eve uses emptying moves only. Suppose that Eve's control state is q_{bB} at the first time when Adam plays $(1, 0)$. Eve's strategy is to play immediately $(q_{bB}, (-1, 0), \top_{bB})$. Because b and B matched so far with the counters, Eve can bring both counters to zero. Note that, if $B = +$, then Eve's strategy is to put B to 0 precisely when the second counter becomes 0. With the subtraction of $e \in \{0, 1\}$ from the first counter in Eve's emptying moves, Eve can also decrement the first counter precisely when Adam plays $(1, 0)$, so she cancels the effect of his move.

As a consequence, when Adam plays $(1, 0)$ whereas b and B match the counters, Eve wins with her strategy.

Hence, if the run of the 2CM returns to a configuration where both counters are 0, then Eve has a winning strategy to reach a configuration with both counters zero from the counter values $(4\bar{y}, \bar{z})$ in the control state $\bar{q}_{\bar{b}\bar{B}}$.

We now prove that Eve loses if the first counter is not a multiple of four after one of her turns. We propose this strategy for Adam, and we will keep the same strategy in the next part of our proof: Adam plays $(1,0)$ whenever (i) b or B does not match the corresponding counter, or (ii) the first counter is a multiple of 4 plus 1 or 2. Otherwise, Adam plays $(0,0)$. To begin with, we point out that the first counter must be a multiple of 4 after each move of Eve. Indeed, if the first counter is not a multiple of 4 after Eve's turn, then at least one of Adam's moves puts the value of the first counter to a multiple of 4 plus 2 or plus 3, and Eve cannot bring back the first counter to a multiple of 4, because the first component of all her moves is either congruent to 3 modulo 4 or a multiple of 4. We show this in Figure 3.9, where we write in square vertices the value modulo 4 of the first counter before a move of Adam, and in circle vertices the value modulo 4 of the first counter before a move of Eve. Arrows indicate Adam's strategy from left to right and any reasonable move of Eve from right to left. We remark that Adam's strategy ensures that the first counter is never again a multiple of 4, as soon as it is not a multiple of 4 after a turn of Eve. In particular, the first counter is then never again zero, hence Eve cannot win.

We can now explain the factor 4 in y: The winner of the game is fixed as soon as the least two significant bits of the first counter are modified, because this means that Adam has played $(1,0)$ or Eve has brought the value of the first counter to a multiple of 4 plus 3. We deduce that the least two significant bits of the first counter are in some sense independent of the rest of the counter, and act as a separate counter.

Finally, let us prove that the strategy that we just proposed for Adam is winning if the instance of Reach-2CM is negative. We make a case split and consider any possible strategy of Eve:

- Suppose that Eve always uses her regular moves and that she maintains the matching between the counters and the flags b and B. It amounts to simulate an infinite run on (Q,T) and, because Adam always plays $(0,0)$, the counter vector cannot become $(0,0)$, in the play like in the 2CM.

- Suppose that Eve always uses her regular moves, but at some point, one of the counters and the corresponding flag of Eve's control state do not match. At this point, because Adam plays $(1,0)$ according to his strategy, there are two possibilities:

  - Eve plays immediately the move $(q_{bB}, (-1,0), \top_{bB})$, where q_{bB} is her internal state. After that, she cannot make a counter value reach zero or stay at zero when the matching does not hold, because she cannot decrease a counter when the corresponding flag is 0 and she must decrease a counter when the corresponding flag is $+$.

  - Eve plays another move, then the first counter is not a multiple of 4 and, as we have explained in the previous part of our proof, Adam's strategy prevents the first counter to be a multiple of 4 ever again.

- Suppose that Eve uses at some point an emptying move, whereas Adam always played $(0,0)$ so far. In this case too, the first counter is not a multiple of 4 and Adam's strategy is winning.

Anyway, Adam's strategy, which is to play $(1,0)$ whenever (i) b or B does not match the

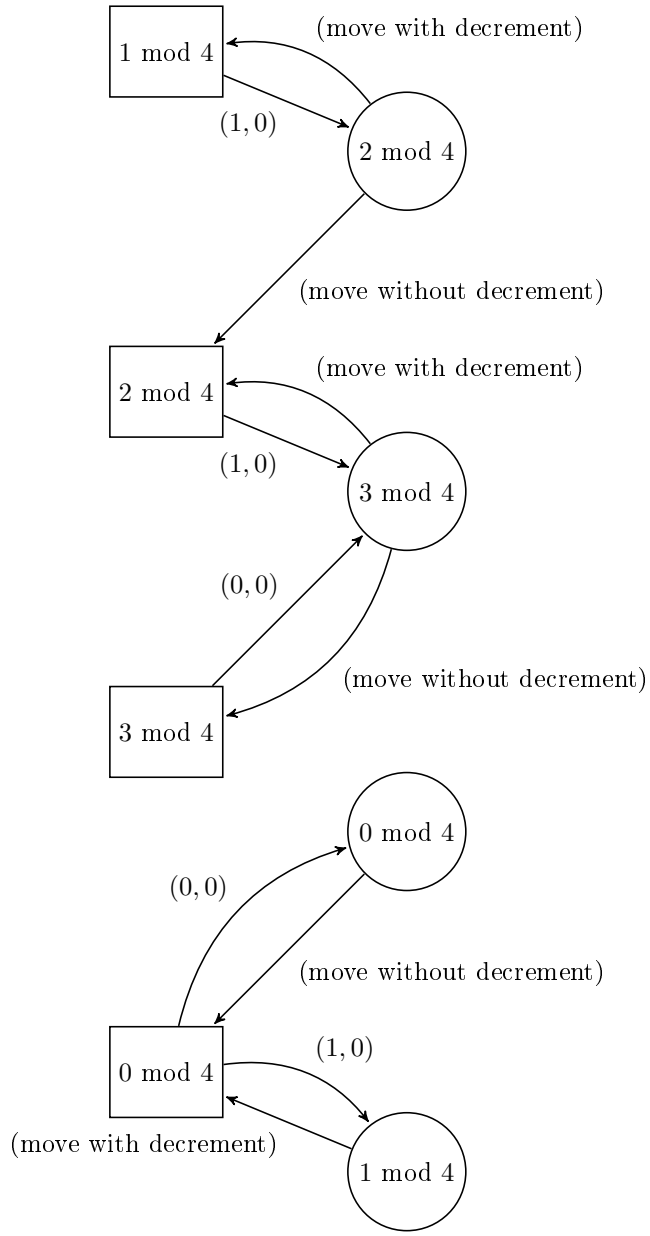

Figure 3.9 – First counter modulo four according to the moves of Adam and Eve.

corresponding counter, or (ii) the first counter is a multiple of 4 plus 1 or 2, and to play $(0,0)$ otherwise, is winning when the instance of Reach-2CM is negative. □

Corollary 45: *The problem of determining the winner of a play in a robot game with states is undecidable in dimension two.*

3.3.3 Second step: Reduction from Reach-2CM to robot games with multiple counters

In the last section, we reduced Reach-2CM to the problem of determining the winner in the new model of two-dimensional robot games with states. We now consider the usual model of robot games, but without fixing the dimension. We make here again a reduction from Reach-2CM, and this time, the dimension of the robot game depends on the Minsky machine in input.

For this second step, we introduce our main feature: the encoding of states into additional counters. First, we explain our reduction and we give the sets A and E of moves of both players. Second, we present an algorithm that extracts from the run of a Minsky machine a strategy for Eve in the robot game (A,E). The strategy that our algorithm extracts is winning if, and only if, the run is accepting. Finally, we prove that our algorithm is correct by explaining why the strategy that we extract is optimal for Eve.

The reduction

Let us consider a 2CM (Q,T) with m states and without self-loop, and a configuration $(q,(y,z))$ such that the run of (Q,T) starts with $(q_0,(0,0))(q,(y,z))\dots$. We build from (Q,T) a robot game (A,E) in dimension $4m+9$. Two counters in (A,E) represent the counters of (Q,T), and each of the $n := 4m+7$ other counters in (A,E), which we call *state counters*, represents one control state of Eve in the extended robot game from the proof of Theorem 44. We prove that Eve has a winning strategy in (A,E) if, and only if, a configuration with vector $(0,0)$ is reachable in (Q,T) from $(q,(y,z))$.

Let us explain how we design the set of state counters that correspond to what would be, up to a modification of sink states, the set of control states of Eve in the first reduction. We define our set Q' of state counters as $Q\times\{0,+\}^2 \cup Q_\top$, where $Q_\top = \{\top_{++}, \top_{++'}, \top_{+0}, \top_{+0'}, \top_{0+}, \top_{0+'}, \top_{00}\}$. The difference between Q' and $(Q\cup\{\top\})\times\{0,+\}^2$ is that we have three more states, $\top_{++'}$, $\top_{+0'}$ and $\top_{0+'}$, to avoid self-loops. We think of the elements of Q' as integers between 1 and n, and we decide that $\top_{++} = n, \top_{++'} = n-1, \top_{+0} = n-2, \top_{+0'} = n-3, \top_{0+} = n-4, \top_{0+'} = n-5$ and $\top_{00} = 1$. The state counters in $Q_\top$ are *emptying counters* and the others are *2CM state counters*. The order of the counters is depicted in Figure 3.10. Like in the previous subsection, we denote an element $(q,(b,B))$ of Q' by q_{bB}. We also keep the function $\delta : \mathbb{N}\to\{0,+\}$ and the notion of *matching* between a counter value and an element of $\{0,+\}$.

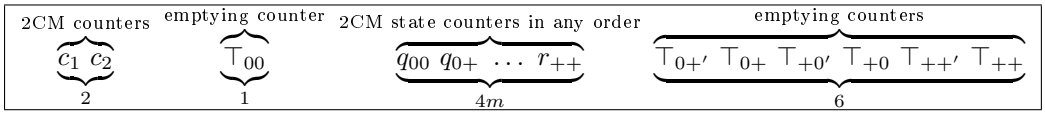

Figure 3.10 – Counters in the $(n+2)$-dimensional vector of the robot game.

Like in Chapter 2, we write in this section $(x)_i^d$, where $x \in \mathbb{Z}$ and $1 \leq i \leq d$, for the d-dimensional vector with x in the i^{th} component and 0 in the other components. We also write 0^d instead of $(0)_1^d$. Note that 8^n, for example, corresponds to 8 to the power of n and is not related to our new notation.

We give names for update vectors that we often use, and we show their effect graphically in Figure 3.11:

- $\textsc{Add}(i,x) := (x)_i^{n+2}$, for $1 \leq i \leq 2$, adds x to the i^{th} counter.
- $\textsc{Move}(j,k) := (-1)_{j+2}^{n+2} + (1)_{k+2}^{n+2}$, for $1 \leq j,k \leq n$, decrements the j^{th} state counter and increments the k^{th} state counter.
- $\textsc{Check}(i) := (-5)_{n-i+3}^{n+2}$, for $1 \leq i \leq 6$, subtracts 5 from the emptying counter $n-i+1$.

0 1 − 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
→ addition of $\textsc{Add}(2,1)$
0 2 − 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
→ addition of $\textsc{Move}(2,5)$
0 2 − 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
→ addition of $\textsc{Check}(1)$
0 2 − 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 -5

Figure 3.11 – Effect of update vectors $\textsc{Add}(2,1)$, $\textsc{Move}(2,5)$ and $\textsc{Check}(1)$ on a counter vector where $m = 3$.

We now come to the construction of the robot game (A,E). For our instance of Reach-2CM, define $\bar{b} = \delta(y)$ and $\bar{B} = \delta(z)$. The initial vector in the robot game (A,E), in dimension $n+2$, is $(4y, z, 0, \ldots, 0) - \textsc{Move}(q_{\bar{b}\bar{B}}, 1)$.

The set of moves A of Adam in the robot game contains what we name *Adam's regular move* $\{0^{n+2}\}$, the *positivity-check* $\{\textsc{Add}(1,1)\}$ and the *state-checks* $\{\textsc{Check}(i), 1 \leq i \leq 6\}$. The set of moves E of Eve contains four types of moves.

First type, Eve's *regular moves*:

- For each transition $(r, c_1\texttt{++}, s) \in T$:
  four moves $\textsc{Add}(1,4) + \textsc{Move}(r_{bB}, s_{+B})$, for $b, B \in \{0,+\}$;
- For each transition $(r, c_1\texttt{--}, s) \in T$:

four moves $\textsc{Add}(1,-4) + \textsc{Move}(r_{+B}, s_{bB})$, for $b, B \in \{0,+\}$;

- For each transition $(r, c_1 == 0, s) \in T$: two moves $\textsc{Move}(r_{0B}, s_{0B})$, for $B \in \{0,+\}$;
- For each transition $(r, c_2\texttt{++}, s) \in T$:
four moves $\textsc{Add}(2,1) + \textsc{Move}(r_{bB}, s_{b+})$, for $b, B \in \{0,+\}$;
- For each transition $(r, c_2\texttt{--}, s) \in T$:
four moves $\textsc{Add}(2,-1) + \textsc{Move}(r_{b+}, s_{bB})$, for $b, B \in \{0,+\}$ (see Figure 3.13);
- For each transition $(r, c_2 == 0, s) \in T$: two moves $\textsc{Move}(r_{b0}, s_{b0})$, for $b \in \{0,+\}$.

With her regular moves, Eve simulates a run of the 2CM, like in the previous reduction.

Second type, Eve's *state-defence moves*:

- $\{\textsc{Move}(r_{bB}, k) - \textsc{Check}(i) \mid b, B \in \{0,+\}, k \in \{\top_{bB}, \top_{bB'}\}, k \neq n+1-i, 1 \leq i \leq 6\}$;
- $\{\textsc{Add}(1,-4e_1) + \textsc{Add}(2,-e_2) - \textsc{Check}(i), e_1, e_2 \in \{0,1\}, 1 \leq i \leq 6\}$;
- $\{\textsc{Add}(1,-4) + \textsc{Add}(2,-1) + \textsc{Move}(j,k) - \textsc{Check}(i),$
$1 \leq i \leq 6, i \neq k, n-1 \leq j \leq n, n-5 \leq k \leq n-2 \vee k = 1\}$;
- $\{\textsc{Add}(1,-4) + \textsc{Move}(j,1) - \textsc{Check}(i), 1 \leq i \leq 6, n-3 \leq j \leq n-2\}$;
- $\{\textsc{Add}(2,-1) + \textsc{Move}(j,1) - \textsc{Check}(i), 1 \leq i \leq 6, n-5 \leq j \leq n-4\}$.

With her state-defence moves, Eve can cancel the effect of any state-check move of Adam. At the same time, she decrements a state counter and increments an emptying counter that is not the one that Adam modified with his state-check move. The possible updates of state counters are either a decrement of a 2CM state counter and an increment of an emptying counter with the same flags b and B, or a decrement of an emptying counter and an increment of another emptying counter, such that no flag changes from 0 to +. This is summed up in Figure 3.12.

Third type, Eve's *emptying moves*, which are defensive moves as well:

- $\{\textsc{Add}(1,-1) + \textsc{Move}(r_{bB}, \top_{bB}) \mid b, B \in \{0,+\}, r \in Q\}$ (see Figure 3.14);
- $\{\textsc{Add}(1,-4-1) + \textsc{Add}(2,-1) + \textsc{Move}(j,k)$
$\mid j \in \{n-1, n\}, k \in \{n+1-i, 1 \leq i \leq 6\} \cup \{1\}, j \neq k\}$;
- $\{\textsc{Add}(1,-4-1) + \textsc{Move}(j,k) \mid j \in \{n-3, n-2\}, k \in \{1, n-3, n-2\}, j \neq k\}$;
- $\{\textsc{Add}(1,-1) + \textsc{Add}(2,-1) + \textsc{Move}(j,k) \mid j \in \{n-5, n-4\}, k \in \{1, n-5, n-4\}, j \neq k\}$;
- $\{\textsc{Add}(1,-4) + \textsc{Add}(2,-1) + \textsc{Move}(j,k)$

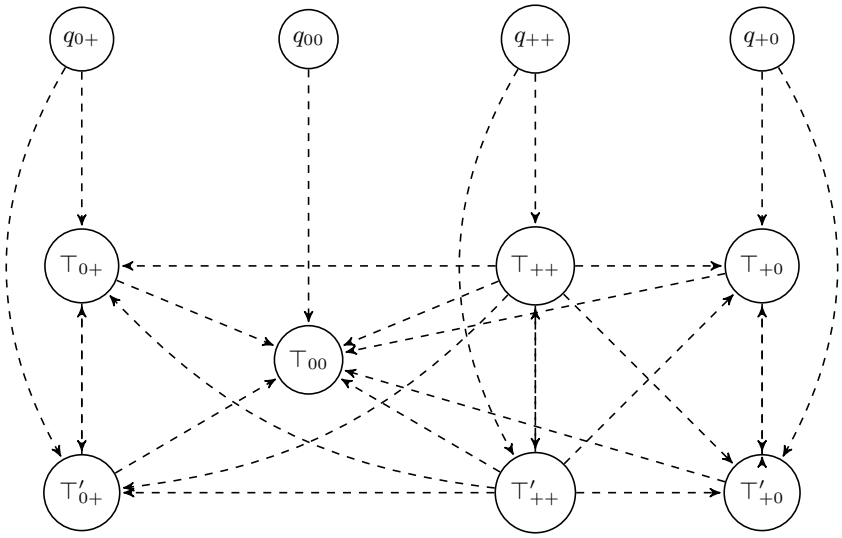

Figure 3.12 – Possible updates of state counters in defence moves: arrows go from the identifier of the state counter that is decremented to the identifier of the state counter that is incremented.

$\mid j \in \{n-1, n\}, k \in \{n+1-i, 1 \leq i \leq 6\} \cup \{1\}, j \neq k\}$;

- $\{\textsc{Add}(1,-4) + \textsc{Move}(j,k) \mid j \in \{n-3, n-2\}, k \in \{1, n-3, n-2\}, j \neq k\}$;
- $\{\textsc{Add}(2,-1) + \textsc{Move}(j,k) \mid j \in \{n-5, n-4\}, k \in \{1, n-5, n-4\}, j \neq k\}$;

With these moves, Eve can bring the 2CM counters to zero. The strategy that we will describe is to use them only when an emptying counter has value 1, though. We distinguish *defence emptying moves*, the first four types of emptying moves, which change the parity of the first counter, and *regular emptying moves*, the last three types of emptying moves. In fact, defence emptying moves cancel the effect of Adam's positivity-check and, like the other emptying moves, they decrease 2CM counters. Only the first defence emptying move stands as an exception: when Eve uses the first defence emptying move, she decrements a 2CM state counter and she increments an emptying counter, besides cancelling the effect of Adam's positivity check.

Fourth type, Eve's *final moves*: $\{\textsc{Move}(j,1), 2 \leq j \leq n\}$. Any of these moves increments the first state counter and decrements another state counter. Recall that the first state counter has value -1 at the beginning of the play.

The algorithm

We explain how to build a strategy for Eve from the run ρ in the 2CM, that is winning if ρ reaches a configuration in $Q \times \{(0,0)\}$. In our strategy, Eve simulates ρ as long as Adam plays his regular

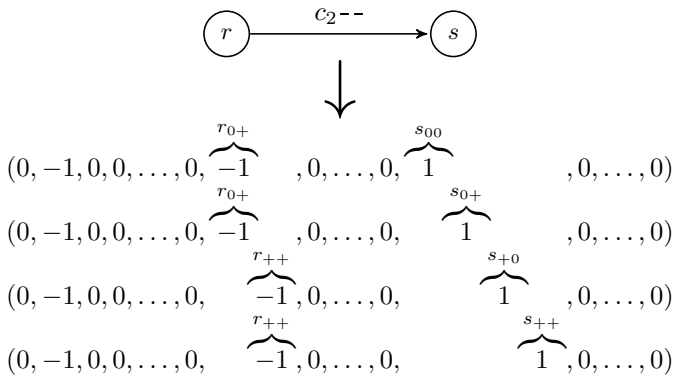

Figure 3.13 – Eve's moves that correspond to the 2CM transition $(r, c_2\texttt{--}, s)$

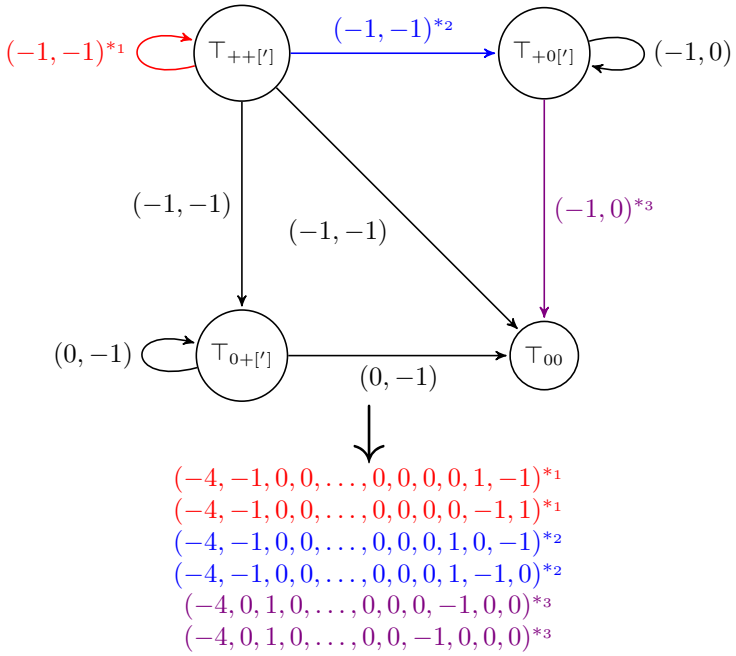

Figure 3.14 – Eve's emptying moves explained on a graph, self-loops are in fact two moves between the two versions of the simulated control state.

move. Meanwhile, Eve also updates the elements of $\{0, +\}$ by incrementing and decrementing relevant state counters, according to whether the 2CM counters are positive or zero. In fact, she maintains the matching between the 2CM counters and the flags b and B of q_{bB}, where q is the state of the 2CM in the corresponding step of the run. Maintaining the matching between the 2CM counters and the flags amounts to ensuring that the only state counter with value 1 is q_{bB}. This is the *simulation phase* of a play. Let us assume that ρ is an accepting run of the 2CM. We prove that it implies that in any play of the robot game, depending on Adam's strategy:

- either Adam always plays his regular move, and consequently the 2CM counters eventually both become zero;
- or Adam does a check at one of the steps.

In the first case, the counter vector becomes $(0, 0, -1, 0, \ldots, 0, 1, 0, \ldots, 0)$, for an arbitrary position of the 1 among the 2CM state counters, so Eve's strategy is to play a final move with which she wins. In the second case, the play enters the *emptying phase*. In the emptying phase, after each turn of Eve, there are exactly two state counter with value $\neq 0$: the counter $\top_{00}$ has value -1 and another emptying counter has value 1. Also, in the emptying phase, each move of Eve depends on the last move a of Adam and cancels the effect of a, while the first two counters are decreased until zero. This phase also ends with a final move or a move that increments the first state counter, so that the counter vector becomes zero. Hence, in both cases, Eve wins with the strategy that we propose.

In the description of the strategy and in our proof, we have some recurring notions. It is expected that Eve simulates the run of a Minsky machine and that Adam plays $(0, 0)$ all along the play. If any of the players does anything else, we call that a *wrong move*. We will see that both Adam and Eve own punishing moves to win when the adversary makes a wrong move, or abuses a punishing move, according to a principle of attack and defence or counterattack.

We sum up Eve's strategy in Algorithm 4, which gives an informal procedure. In particular, at every step, the algorithm contains the instruction "Play" associated with a vector, which corresponds to the strategy at the current position in the play. In fact, the input of the algorithm is not only the 2CM and its run ρ, but also a strategy of Adam, and the instructions "Play" give the strategy on the fly, one move after the other and depending on Adam's moves.

The variables that appear in the algorithm are the counter vector x in the robot game, a variable to keep the index of the current step of ρ, the corresponding state p in ρ, the variables b and B for the flags such that the state counter p_{bB} has value one in x, at least during the simulation phase.

Algorithm 4 calls a function `move_of_transition`, shown in Algorithm 5. The function `move_of_transition` gives Eve's move that corresponds to a transition in the 2CM, depending on the counter values. The variables of the main algorithm are updated during the execution of the function, and we recall this in the algorithm by pointing out the existence of side effects upon calling `move_of_transition`. Indeed, all variables are global, except the index of the step of ρ. The choice of the move is intuitive, for example the transition $(r, c_2\texttt{--}, s)$ can only be fired if the

second counter is positive, so the flag B must be $+$, and after a decrement the second counter is either zero, if the value was exactly one before, or still positive. Accordingly, there is a test on the value of the second counter, and the possible moves that correspond to $(r, c_2\texttt{--}, s)$ are either $\textsc{Add}(2,-1) + \textsc{Move}(p_{b+}, s_{b+})$ or $\textsc{Add}(2,-1) + \textsc{Move}(p_{b+}, s_{b0})$.

Whereas the calls to the function `move_of_transition` deal with the simulation phase of a play, the remaining part of Algorithm 4 relates to cases where Adam performs a check. The first check of Adam is a particular move: according to her strategy, Eve only used regular moves before, so only one state counter has value one, and this counter is a p_{bB}, for $p \in Q$. According to the strategy described in the algorithm, Eve cancels the effect of Adam's check and makes a move to $\top'_{bB}$ if Adam's check is precisely the move $\textsc{Check}(i)$ such that $\top_{bB}$ is the i^{th} state counter, and to $\top_{bB}$ otherwise. After the first defence move of Eve, only state counters with state $\top$ are updated, and Eve cancels the effect of each move of Adam and decrements the first two counters until they become zero. Meanwhile, she ensures that the flags of the state counter that has value one match the counter values, which corresponds to the instructions "if $x[0] == 4$ then $b \leftarrow 0$" and "if $x[1] == 1$ then $B \leftarrow 0$". To sum up, Eve uses defence moves until the whole counter vector becomes zero.

The proof of correctness

The following lemmas cover all possible cases in a play on (Q', T'), and prove that Eve has a winning strategy if, and only if, the run of the Minsky machine is accepting. The strategy that we prove to be winning is exactly the strategy that we present in Algorithm 4: it consists of a simulation of the run of the Minsky machine in the simulation phase and of appropriate defence if Adam attacks, because an attack of Adam can only be a wrong move.

In our proof of correctness, in the first four lemmas, we point out invariants that Eve needs to preserve in any play, otherwise Adam would have a winning move. After these lemmas, we assume that Eve always preserves the invariants, which restricts her possible strategies. We then prove, by studying all possible strategies of Adam, that:

- If Eve has a winning strategy, then the strategy that we present in Algorithm 4 is winning.
- Eve has a winning strategy if, and only if, the instance of Reach-2CM is positive.

Lemma 46: *Consider an arbitrary play on (A, E), where Adam and Eve play to the best of their interest. If Adam never plays his positivity-check move and Eve plays a defence emptying move, then Eve loses the play. Moreover, when Adam plays his positivity-check move for the first time, if Eve does not play directly afterwards a defence emptying move, then she loses the play.*

Proof. Note that defence emptying moves are Eve's only moves that modify the parity of the first counter. If the least two significant bits of the binary representation of the first counter are not zero after a turn of Eve, then Adam has a strategy to prevent these bits to become zero again, hence he wins: he plays the positivity-check move at every turn when these two bits are not 1 and 1.

Algorithm 4: Extracts from the run of a 2CM, supposed to be accepting, a winning strategy for Eve in the robot game constructed in the reduction from Reach-2CM.

Input: A 2CM (Q,T), the run ρ of (Q,T) supposed to be accepting and given as a list of transitions, the second configuration $(q,(y,z))$ of ρ, a strategy of Adam.
Result: A winning strategy for Eve in the game obtained from (Q,T).
begin
  $b \leftarrow \delta(y)$; $B \leftarrow \delta(z)$; $p \leftarrow q$
  $x \leftarrow (4y, z, 0, \ldots, 0) - \textsc{Move}(q_{bB}, 1)$ /* counter vector */
  step $\leftarrow 0$ /* step in the run ρ */
  while *Adam plays* 0^d **do**
    if $x[0] == 0 \wedge x[1] == 0 \wedge step \neq 0$ `/* the first two counters are zero, but not in the beginning */` **then**
      Play $\textsc{Move}(p_{bB}, 1)$ `/* at this point, Eve wins */`
    else
      $v \leftarrow$ move_of_transition(ρ[step]) `See Algorithm 5 for side effects`
      Play v
      step++
  repeat
    $v \leftarrow$ Adam's move
    if $p \in Q$ `/* the first move in this block */` **then**
      if $-v + \textsc{Move}(p_{bB}, \top_{bB}) \in E$ `/* when` v `is a` Check`, the test may be false */` **then**
        $x \leftarrow x + \textsc{Move}(p_{bB}, \top_{bB})$; $p \leftarrow \top_{bB}$
        Play $-v + \textsc{Move}(p_{bB}, \top_{bB})$
      else
        $x \leftarrow x + \textsc{Move}(p_{bB}, \top_{bB'})$; $p \leftarrow \top_{bB'}$
        Play $-v + \textsc{Move}(p_{bB}, \top_{bB'})$
    else
      if $b == +$ **then** $d_1 \leftarrow \textsc{Add}(1, -4)$
      else $d_1 \leftarrow 0$
      if $B == +$ **then** $d_2 \leftarrow \textsc{Add}(2, -1)$
      else $d_2 \leftarrow 0$
      if $x[0] == 4$ **then** $b \leftarrow 0$
      if $x[1] == 1$ **then** $B \leftarrow 0$
      $d_3 \leftarrow$ (a move, possibly 0, from p to $\top_{bB}$ or $\top_{bB'}$, depending on which ones are in E)
      $x \leftarrow x + d_1 + d_2 + d_3$
      Play $-v + d_1 + d_2 + d_3$
  until *the end of the play*

Algorithm 5: move_of_transition

Input: A transition t in a run ρ.
Result: The move according to Eve's strategy in the simulation phase.
begin

```
/* Updates of the variables are applied as side effects in the main
algorithm */
```

if $t == (r, c_1\texttt{++}, s)$ `/*` $p = r$`, state:` r_{bB} `*/` **then**

$x \leftarrow x + (4)_1^d + \text{MOVE}(p_{bB}, s_{+B})$; $b \leftarrow +$; $p \leftarrow s$
Return $\text{ADD}(1, 4) + \text{MOVE}(p_{bB}, s_{+B})$

else if $t == (r, c_2\texttt{++}, s)$ `/*` $p = r$`, state:` r_{bB} `*/` **then**

$x \leftarrow x + (1)_2^d + \text{MOVE}(p_{bB}, s_{b+})$; $B \leftarrow +$; $p \leftarrow s$
Return $\text{ADD}(2, 1) + \text{MOVE}(p_{bB}, s_{b+})$

else if $t == (r, c_1\texttt{--}, s)$ `/*` $p = r$`, state:` r_{+B} `*/` **then**

if $x[0] == 4$ **then** $b \leftarrow 0$ `/* side effect */`

$x \leftarrow x - (4)_1^d + \text{MOVE}(p_{+B}, s_{bB})$; $p \leftarrow s$
Return $\text{ADD}(1, -4) + \text{MOVE}(p_{+B}, s_{bB})$

else if $t == (r, c_2\texttt{--}, s)$ `/*` $p = r$`, state:` r_{b+} `*/` **then**

if $x[1] == 1$ **then** $B \leftarrow 0$ `/* side effect */`

$x \leftarrow x - (1)_2^d + \text{MOVE}(p_{b+}, s_{bB})$; $p \leftarrow s$
Return $\text{ADD}(2, -1) + \text{MOVE}(p_{b+}, s_{bB})$

else if $t == (r, c_1 \texttt{==} 0, s)$ `/*` $p = r$`, state:` r_{0B} `*/` **then**

$x \leftarrow x + \text{MOVE}(p_{0B}, s_{0B})$; $p \leftarrow s$
Return $\text{MOVE}(p_{0B}, s_{0B})$

else if $t == (r, c_2 \texttt{==} 0, s)$ `/*` $p = r$`, state:` r_{b0} `*/` **then**

$x \leftarrow x + \text{MOVE}(p_{b0}, s_{b0})$; $p \leftarrow s$
Return $\text{MOVE}(p_{b0}, s_{b0})$

Note that, as we consider that Adam and Eve play to the best of their interest, even though it is not required that Adam plays indeed according to the strategy that we propose in the previous paragraph, it is guaranteed that Adam plays according to a winning strategy when the least two significant bits of the binary representation of the first counter are not zero after a turn of Eve.

Here, the wrong move is either the positivity check, or the defence emptying move, or an earlier move of the game. In any case, the nature of the wrong move is less important than the parity of the counters, and we make the link with the expected behaviours in a later lemma. □

Lemma 47: *Consider an arbitrary play on (A, E), where Adam and Eve play to the best of their interest. If Adam never plays a state-check move and Eve plays a state-defence move, then Eve loses. Moreover, when Adam plays a state-check move, if Eve does not play directly afterwards a corresponding state-defence move, then she loses.*

Proof. The sum of the state counters is zero at the beginning of a play: the counter $\top_{00}$ has value -1 and the counter $q_{\bar{b}\bar{B}}$ has value 1. Every move apart from the state-check and state-defence moves leaves the sum of the state counters unchanged, whereas the state-check moves increase the sum of the state counters by five and the state-defence moves decrease the sum of the state counters by five. Hence, suppose that Eve plays something else than a state-defence move after a state-check move of Adam. Adam can then guarantee that the sum of the state counters is never zero again by playing only state-check moves. Moreover, if the state-defence move of Eve does not correspond to the state-check move of Adam, then there will be a state counter with non-zero value after each following turn. As a consequence, the counter vector cannot become zero, hence Eve loses.

Like in the proof of the previous lemma, the wrong move is either the state-check, or the state-defence move, or an earlier move of the game. Still, what really matters is a property that we point out: Eve needs to ensure that the sum of the state counters remains zero after all her turns. □

Note that Lemma 47 does not hold if the value of the state counter that the state-check move modifies is for example 5, but in this case Adam should not use this move at all.

Lemma 48: *Consider an arbitrary play on (A, E), where Adam and Eve play to the best of their interest. If a state counter other than the first one is negative after any turn of Eve, then she loses the play.*

Proof. First, we prove that Eve loses if a 2CM state counter is negative after one of her turns. A 2CM state counter can only be increased, namely incremented, by Eve's regular moves. Hence, if one of the 2CM state counters becomes negative, then Adam wins by playing state-check moves only, because according to Lemma 47, Eve then must play state-defence moves only and cannot bring back a negative 2CM state counter to zero. Second, suppose that an emptying counter $n + 1 - i$, for $1 \leq i \leq 6$, is negative after a turn of Eve. Adam then wins by playing $\textsc{Check}(i)$ forever, because Eve can increase a state counter by at most five at each turn, which just compensates a move $\textsc{Check}(i)$ but is not sufficient to make the counter $n + 1 - i$ nonnegative again. □

As a consequence, since Eve's regular moves decrement a 2CM state counter and only one 2CM state counter has value one, we have a new invariant: during the simulation phase, exactly one 2CM state counter has value 1, the counter $\top_{00}$ has value -1 and all other state counters have value 0; moreover, Eve must decrement at every turn the 2CM state counter that has value 1, otherwise she loses the play.

Lemma 49: *Consider an arbitrary play on (A, E), where Adam and Eve play to the best of their interest. We suppose that exactly one state counter has value one after every turn of Eve during the simulation phase. We consider the flags, in $\{0, +\}$, of the identifier of the only state counter that has value one. If one of the first two counters does not match the corresponding flag in $\{0, +\}$ after any turn of Eve, then she loses the play.*

Proof. We assume that Eve decrements at every turn the state counter that has value 1 during the simulation phase. As we stated after the previous lemma, Eve would lose otherwise.

Let us denote by p_{bB} the identifier of the state counter that has value 1 at some step of the play. If one of the first two counters does not match the corresponding flag b or B, then Adam can play his positivity-check move, which forces Eve to play a defence emptying move, according to Lemmas 46 and 48. At this point, the matching will still not hold, because the flags and the first two counters remain unchanged. Let us look at the emptying moves. Eve cannot decrease a counter if the corresponding flag is 0 and she decreases it if the corresponding flag is $+$. Consequently, Adam wins by playing the positivity-check move forever, because the first two counters cannot be both brought to zero at the same time. Indeed, Eve needs to play defence emptying moves at all steps, still according to Lemma 46. Now, there are two possibilities:

- Either the matching does not hold because one of the first two counters, say the second counter, is 0 whereas the corresponding flag is $+$. Then, the next defence emptying move of Eve decrements the second counter, and none of the further defence emptying moves increases the second counter again, so Eve cannot bring the counter vector to zero: she loses the play.

- Or the matching does not hold because one of the first two counters, say the second counter, is positive whereas the corresponding flag is 0. Then, the same flag will remain at zero after all further defence emptying moves, none of which decrease the second counter, which will never be 0: Eve loses the play.

□

Lemma 50: *Let ρ be the run of (Q, T) and suppose that it reaches a configuration in $Q \times \{(0, 0)\}$. Consider a play on (A, E), in which Eve follows the strategy based on ρ and described in Algorithm 4. If Adam plays at some step his positivity-check move and never plays any state-check move, then Eve wins.*

Proof. Eve maintains the matching, we can see it in the function `move_of_transition` of Algorithm 5. Suppose that Adam plays his positivity-check move and that he never plays any state-check move. Eve can then make every counter zero with her emptying moves, precisely because

the matching holds. The argument is the same as in the reduction from Reach-2CM to extended 2RG. □

Lemma 51: *Let ρ be the run of (Q,T) and suppose that it reaches a configuration in $Q \times \{(0,0)\}$. Consider a play on (A,E), in which Eve follows the strategy based on ρ and described in Algorithm 4. If Adam plays a state-check move, then Eve wins.*

Proof. Suppose that Adam plays a state-check move. According to her strategy, Eve cancels immediately the effect of Adam's move with a corresponding state-defence move. Because of the state-check move, one of the emptying counters cannot be incremented. Nevertheless, there is at least one corresponding state-defence move that gives the value 1 to an emptying counter that corresponds to the positivity of the machine counters. After that, either Adam does not play a state-check move and Eve plays an emptying move, or Adam plays a state-check move and Eve cancels the effect of this move and still decreases the first two counters. This is repeated until the end of the play. In both cases, Eve can put the first two counters to zero and use a move that increments the first state counter afterwards, hence she wins. □

Theorem 52: *Let ρ be the run of (Q,T), denote by $(q,(y,z))$ the second configuration in ρ. Let $\bar{b} = \delta(y)$ and $\bar{B} = \delta(z)$. Eve has a winning strategy in (A,E) from $(4y,z,0,\ldots,0) - \textsc{Move}(q_{\bar{b}\bar{B}},1)$ if, and only if, ρ returns to a configuration in $Q \times \{(0,0)\}$.*

Proof. We still assume that Eve decrements at every turn the state counter that has value 1 during the simulation phase, according to Lemma 48. As we stated after the previous lemma, Eve would lose otherwise. Note that, because of Lemma 49, Eve's regular moves must correspond to transitions that are allowed in the 2CM.

Hence, the strategy described in Algorithm 4, which fulfils this constraint, is winning for Eve against the strategy "only the regular move" of Adam if the run of the 2CM (Q,T) returns to a configuration with counter vector $(0,0)$.

Moreover, according to Lemmas 50 and 51, the strategy is winning against every other strategy of Adam.

For the converse implication, according to Lemma 46 and 47, a strategy of Eve that does not fulfil the above constraint is losing. Hence, the only reasonable strategies of Eve consist of simulating the run ρ of the 2CM and using defence moves as soon as Adam plays another move than his regular move. Indeed, everything else than the simulation of ρ is a wrong move, and we have proved that it makes Eve lose. If ρ does not return to any configuration in $Q \times \{(0,0)\}$, then Eve cannot bring the first two counters to zero with the strategy that consists of simulating ρ, so she does not have any winning strategy at all. □

The following proposition points out an additional property of the strategy that we describe in Algorithm 4. In fact, we need to bound the values of the state counters in anticipation of the next step, where we encode vectors in integers and we must avoid any ambiguity in our encoding.

Proposition 53: *If Eve has a winning strategy in (A, E), then she also has a winning strategy such that all state counters have a value between -5 and 1 after each move.*

Proof. Consider the strategy s described in Algorithm 4. It Eve plays according to s and Adam plays according to any strategy, then the play that is constructed has the following properties:

- The first state counter always has value -1 or 0 and Adam cannot modify it. The other state counters always have value 0 or 1 and Adam can only decrease them by 5 or leave them unmodified.
- The effect of each move of Adam is immediately cancelled, insofar as the strategy returns a move that contains the opposite of Adam's move. Actually, cancelling the effect of Adam's regular move amounts to doing nothing specific.

Hence, Adam can at worst bring a state counter to the value -5, after which Eve increases the counter to 0 again. For the upper bound, Adam cannot increase state counters and Eve's moves never put any state counter to a greater value than 1.

Moreover, we have proved that s is a winning strategy if, and only if, there run of the 2CM is accepting, and if the run of the 2CM is not accepting, then Eve does not have any winning strategy. □

3.3.4 Reduction from Reach-2CM to 3RG

For this final reduction, we use the same principle as in the previous reduction to build a robot game in dimension three from a 2CM. The only difference is that instead of $4m + 7$ state counters we have one counter that stores the same information. In our proof, the focus is on how to avoid undesired side effect while merging counters.

Let $f : \mathbb{Z}^n \to \mathbb{Z}$ be the encoding basis function $(x_1, \ldots, x_n) \mapsto \sum_{i=1}^{n} x_i 8^{i-1}$. We encode n values, which correspond to the state counters in the game (A, E) of the previous section, in an integer with the function f.

We rewrite the update vectors that we often use:

- $\textsc{Add}(1, x) = (x, 0, 0)$ and $\textsc{Add}(2, x) = (0, x, 0)$.
- $\textsc{Move}(i, j) = (0, 0, f((1)^n_j - (1)^n_i))$, for $1 \leq i, j \leq n$.
- $\textsc{Check}(i) = (0, 0, f((-5)^n_{n+1-i}))$, for $1 \leq i \leq 6$.

We reuse the notations (Q, T), $(q, (y, z)), b, B, q_{\bar{b}\bar{B}}$ and Q' from the previous section. Let (A_1, E_1) be a three-dimensional robot game, with initial vector $(4y, z, 8^{n+1} + 8^n + f((1)^n_{q_{\bar{b}\bar{B}}} - (1)^n_1))$

where $A_1 = \{(0,0,0), \textsc{Add}(1,1)\} \cup \{\textsc{Check}(i), 1 \le i \le 6\}$ and E_1 is the following set, for which we emphasize the differences to E:

- – For each transition $(r, c_1\texttt{++}, s) \in T$:
  four moves $\textsc{Add}(1,4) + \textsc{Move}(r_{bB}, s_{+B})$, for $b, B \in \{0,+\}$;

  – For each transition $(r, c_2\texttt{++}, s) \in T$:
  four moves $\textsc{Add}(2,1) + \textsc{Move}(r_{bB}, s_{b+})$, for $b, B \in \{0,+\}$;

  – For each transition $(r, c_1\texttt{--}, s) \in T$:
  four moves $\textsc{Add}(1,-4) + \textsc{Move}(r_{+B}, s_{bB})$, for $b, B \in \{0,+\}$;

  – For each transition $(r, c_2\texttt{--}, s) \in T$:
  four moves $\textsc{Add}(2,-1) + \textsc{Move}(r_{b+}, s_{bB})$, for $b, B \in \{0,+\}$;

  – For each transition $(r, c_1 \texttt{==} 0, s) \in T$: two moves $\textsc{Move}(r_{0B}, s_{0B})$, for $B \in \{0,+\}$;

  – For each transition $(r, c_2 \texttt{==} 0, s) \in T$: two moves $\textsc{Move}(r_{b0}, s_{b0})$, for $b \in \{0,+\}$;

- – $\{\textsc{Move}(r_{bB}, k) - \textsc{Check}(i) + (\mathbf{0}, \mathbf{0}, -\mathbf{8^n})$
  $\mid r \in Q, (b,B) \in \{0,+\}^2 \backslash \{(\mathbf{0},\mathbf{0})\}, k \in \{\top_{bB}, \top_{bB'}\}, k \ne n+1-i, 1 \le i \le 6\}$;

  – $\{\textsc{Move}(r_{00}, 1) - \textsc{Check}(i) + (\mathbf{0}, \mathbf{0}, -\mathbf{8^{n+1}} - \mathbf{8^n}) \mid r \in Q\}$;

  – $\{\textsc{Add}(1, -4e_1) + \textsc{Add}(2, -e_2) - \textsc{Check}(i) \mid e_1, e_2 \in \{0,1\}, 1 \le i \le 6\}$;

  – $\{\textsc{Add}(1,-4) + \textsc{Add}(2,-1) + \textsc{Move}(j,k) - \textsc{Check}(i)$
  $\mid 1 \le i \le 6, i \ne k, 1 \le j \le 2, 3 \le k \le \mathbf{6}\}$;

  – $\{\textsc{Add}(1,-4) + \textsc{Add}(2,-1) + \textsc{Move}(j,1) - \textsc{Check}(i) + (\mathbf{0}, \mathbf{0}, -\mathbf{8^{n+1}})$
  $\mid 1 \le i \le 6, 1 \le j \le 2\}$;

  – $\{\textsc{Add}(1,-4) + \textsc{Move}(j,1) - \textsc{Check}(i) + (\mathbf{0}, \mathbf{0}, -\mathbf{8^{n+1}}) \mid 1 \le i \le 6, 3 \le j \le 4\}$;

  – $\{\textsc{Add}(2,-1) + \textsc{Move}(j,1) - \textsc{Check}(i) + (\mathbf{0}, \mathbf{0}, -\mathbf{8^{n+1}}) \mid 1 \le i \le 6, 5 \le j \le 6\}$;

- – $\{\textsc{Add}(1,-1) + \textsc{Move}(r_{bB}, \top_{bB}) + (\mathbf{0}, \mathbf{0}, -\mathbf{8^n}) \mid (b,B) \in \{0,+\}^2 \backslash \{(\mathbf{0},\mathbf{0})\}, r \in Q\}$

  – $\{\textsc{Add}(1,-1) + \textsc{Move}(r_{00}, \top_{00}) + (\mathbf{0}, \mathbf{0}, -\mathbf{8^{n+1}} - \mathbf{8^n}) \mid r \in Q\}$;

  – $\{\textsc{Add}(1,-4-e) + \textsc{Add}(2,-1) + \textsc{Move}(j,k)$
  $\mid 0 \le e \le 1, j \in \{n-1, n\}, k \in \{n+1-i, 1 \le i \le 6\}, j \ne k\}$;

  – $\{\textsc{Add}(1,-4-e) + \textsc{Add}(2,-1) + \textsc{Move}(j,1) + (\mathbf{0}, \mathbf{0}, -\mathbf{8^{n+1}}) \mid 0 \le e \le 1, j \in \{n-1, n\}\}$;

  – $\{\textsc{Add}(1,-4-e) + \textsc{Move}(j,k) \mid e \in \{0,1\}, j \in \{n-3, n-2\}, k \in \{\mathbf{n-3}, \mathbf{n-2}\}, j \ne k\}$;

- $\{\textsc{Add}(1,-4-e)+\textsc{Move}(j,1)+(\mathbf{0},\mathbf{0},-\mathbf{8^{n+1}}) \mid e\in\{0,1\}, j\in\{n-3,n-2\}\}$;

- $\{\textsc{Add}(1,-e)+\textsc{Add}(2,-1)+\textsc{Move}(j,k)$
  $\mid e\in\{0,1\}, j\in\{n-5,n-4\}, k\in\{\mathbf{n-5},\mathbf{n-4}\}, j\neq k\}$;

- $\{\textsc{Add}(1,-e)+\textsc{Add}(2,-1)+\textsc{Move}(j,1)+(\mathbf{0},\mathbf{0},-\mathbf{8^{n+1}}) \mid e\in\{0,1\}, j\in\{n-5,n-4\}\}$;

- $\{\textsc{Move}(j,1)+(\mathbf{0},\mathbf{0},-\mathbf{8^{n+1}}) \mid j\in\{n+1-i, 1\leq i\leq 6\}\}$
  $\cup\{\textsc{Move}(j,1)+(\mathbf{0},\mathbf{0},-\mathbf{8^{n+1}}-\mathbf{8^{n}}) \mid 2\leq j\leq n-6\}$.

In our construction, the information about the states of the 2CM is in the third counter.

The differences between the moves in the game (A,E) and the moves in the game (A_1,E_1) come from possible carries. Because of carries, two vectors of state counters would give the same value for the third counter, for example $f(4,1,0,\ldots,0)=12=f(-4,2,0,\ldots,0)$. Counter updates may involve carries in the game (A_1,E_1), because state counters may have a value outside $\{0,\ldots,7\}$, for example when Adam performs a state check. Nevertheless, there is no risk of ambiguity because the possible range of the state counters is an interval of length at most eight, namely $\{-5,\ldots,1\}$, according to Proposition 53. Note that we write moves with an $e\in\{0,1\}$ in the third block, to gather defence emptying moves and regular emptying moves for a more concise description of the moves.

In the game (A,E), according to Proposition 53, if Eve has a winning strategy, then she also has a winning strategy s that ensures that the state counters have at every step a value between -5 and 1, hence, a winning strategy, which is adapted from s, in (A_1,E_1).

We now have to prove that Adam also can prevent Eve to use carries to her advantage, concretely, that Adam has a winning strategy in (A_1,E_1) if he has a winning strategy in (A,E). This is the purpose of the most two significant digits of the third counter. We prove that Eve must decrement exactly once each of these two digits: at most once because she cannot make the third counter become negative, at least once because Adam has a punishing strategy if she tries to use carries to decrease the two digits without modifying them with dedicated moves. According to Eve's set of moves, we conclude that Eve can make the counter vector become zero, while indeed decrementing exactly once each one of the most two significant digits, if, and only if, she simulates the run ρ of the 2CM and ρ is accepting.

Lemma 54: *Let (A_1,E_1) be the three-dimensional robot game that we build by our last reduction from Reach-2CM. Consider an arbitrary play on (A_1,E_1), where Adam and Eve play to the best of their interest. If, after any turn of Eve, the third counter is negative, then Eve loses the play.*

Proof. Let us prove that, whenever the third counter becomes negative, Adam wins if he then plays $\textsc{Check}(1)$ on and on.

Proposition 25 on page 59 states that, in a robot game in dimension one, if the least move of Adam is smaller or equal to the opposite of the greatest move of Eve, then no negative value

is winning for Eve. This still holds, analogously, in any dimension: if the lowest value in the d^{th} dimension of the moves of Adam is smaller than or equal to the opposite of the greatest value in the d^{th} dimension of the moves of Eve, then no counter vector with a negative d^{th} component is winning for Eve.

Here, the lowest value in the third dimension of the moves of Adam is CHECK(1) and the greatest value in the third dimension of the moves of Eve is $-$CHECK(1), which increases the third most significant digit. Let us consider all moves of Eve that contain $-$CHECK(1) in their description. Some of these moves also decrement at least one of the most two significant digits, others contain a move that also increments a digit and decrements a more significant digit, all others contain no MOVE at all. In any case, the sum of Adam's move CHECK(1) and of any move e of Eve, with or without $-$CHECK(1) in the description of e, has a negative component in the third dimension. To sum up, when Adam plays CHECK(1) and Eve plays any of her moves, the third dimension of the counter vector is not increased. Hence, if the third dimension of the counter vector becomes negative, then Adam wins by playing CHECK(1) forever. □

Lemma 55: *Let (A_1, E_1) be the three-dimensional robot game that we build by our last reduction from Reach-2CM. Consider an arbitrary play on (A_1, E_1), where Adam and Eve play to the best of their interest. During the play, if Eve does not use exactly once either a move that contains $(0, 0, -8^{n+1} - 8^n)$ in its description or a move that contains $(0, 0, -8^{n+1})$ in its description and a move that contains $(0, 0, -8^n)$ in its description, then she loses.*

Proof. First, we deal with the most significant digit of the third counter. We find constraints on Eve's moves to prove that she needs to decrement the most significant digit exactly once. Second, we deal with the second most significant digit of the third counter.

The most significant digit of the third counter is decremented exactly by Eve's moves that increment the least significant digit of the third counter, and there is no other possible update of the least significant digit than the increments. Incidentally, Eve cannot increment the most significant digit of the third counter. As the least significant digit is non-zero at the beginning of a play, Eve must use a move that modifies the two end digits at least once. Moreover, using such a move more than once would decrement the most significant digit when it is already zero, so the third counter would become negative. Then, according to Lemma 54, Adam would win. To sum up, Eve must decrement the most significant digit of the third counter exactly once.

The second most significant digit of the counter is decremented by every move MOVE(i, j) of Eve such that i corresponds to a state of the 2CM and j corresponds to a state counter. If Eve never uses such a move, then the digits that correspond to the states of the 2CM cannot become all zero a priori. In fact, there are two possibilities to decrement a digit: either a direct decrement, or a carry caused by a decrease of a lower digit.

Let us rule out that Eve does a direct decrement of the second most significant digit of the third counter, by never using a move of the form that we give in the beginning of the paragraph. If Eve manages to decrease in (A_1, E_1) the second most significant digit with a carry, it necessarily corresponds in (A, E) to a state counter that she made become negative. Then, Adam can attack

on the corresponding digit of the third counter with CHECK moves, so that at least one lower digit of the third counter never becomes zero again, hence the third counter cannot become zero, and Eve loses. We conclude that Eve must do a direct decrement of the second most significant digit of the third counter, possibly with the same move as the move with which she decrements the most significant digit of the third counter. □

Theorem 56: *Let ρ be the run of (Q,T), denote by $(q,(y,z))$ the second configuration in ρ. Let $\bar{b}=\delta(y)$ and $\bar{B}=\delta(z)$. Eve has a winning strategy in (A_1,E_1) from $(4y,z,8^{n+1}+8^n)-\text{MOVE}(q_{\bar{b}\bar{B}},1)$ if, and only if, ρ returns to a configuration with counter value $(0,0)$.*

Proof. The strategy is exactly the same as in the game (A,E), and Lemma 55 ensures that Adam can prevent Eve from creating ambiguous configurations with carries on the third counter, whereas there is no ambiguity in (A,E). Therefore, if Eve has a winning strategy in (A_1,E_1), then she also has one in (A,E) and conversely. □

Corollary 57: *The problem of determining the winner of a robot game in dimension three is undecidable.*

Unfortunately, the technique that we use in this section does not seem to be extendable to dimension two.

Chapter 4

A Graphical Tool for Representing the Winning Set in Two-Dimensional Robot Games

An open problem remains, in light of the work in the chapter about robot games: what about dimension two? Whereas it is believable that the proof of undecidability can be adapted with two counters only, nothing seems to work when we try to merge an unbounded counter with a bounded one, because the property that Eve loses whenever the bounded counter becomes negative (Lemma 54) is fundamental and would no longer hold if we used a naive concatenation.

Indeed, handling possible carries between different parts of a counter is not obvious, as we can see in the reductions from countdown games to the problem of determining the winner of a one-dimensional robot game and from the reachability problem on Minsky machines to the problem of determining the winner of a three-dimensional robot game. In the two reductions that we mention, we managed to prove that carries did not cause any trouble in the robot games that we build, but all counters that we merged and encoded in one counter were meant to be bounded, which would not be the case if we tried to extend the proof of undecidability to dimension two, in other words, if we merged the third counter with one of the counters of the 2CM, necessarily in the least significant digits because the counters of the 2CM are unbounded. In some sense, Eve needs to have moves with much variety so that she can defend against Adam's attacks, and the problem lies in the difficulty to prevent such moves to have an effect on the above part of the merged counter, i.e., the value of a 2CM counter. If there were any possible effect, due to a carry, on the value of a 2CM counter in a merged counter, then we could have examples of ambiguous two-dimensional counter vectors corresponding to different three-dimensional counter vectors and still obtainable by Eve in the two-dimensional robot game, which is not acceptable.

In short, it is still possible, but delicate, to look for a proof that the problem of determining the

winner of a two-dimensional robot games is undecidable, but it is possible as well that the problem of determining the winner of a two-dimensional robot game is decidable after all. Accordingly, to investigate on decidability, or at least for particular cases, I implemented a graphical tool[1] that computes the attractor on $\mathbb{Z} \times \mathbb{Z}$.

With this tool, it is possible to have an overview of the winning set in a robot game, and moreover to see how it is computed step by step, i.e., from which counter vectors Eve has a strategy to win in at most a given number of moves. Of course, the tool is in no way an algorithm, because termination is not guaranteed and would not be guaranteed either for Algorithm 3, which computes Eve's winning set in a one-dimensional robot game, if we did not stop the iterations, using arithmetical results presented beforehand, while computing the attractor. The reason why the tool was written is precisely because finding out a stopping criterion in dimension two could be easier with an illustration.

Among the attempts to find decidable classes of robot games in dimension two, sometimes supported by hypotheses that the tool helped to formulate, we present the main extensions with positive results:

- *Only horizontal or vertical moves*: Adam and Eve have moves in $\mathbb{Z} \times \{0\} \cup \{0\} \times \mathbb{Z}$;
- *unary moves*: Adam and Eve have moves in $\{-1, 0, 1\} \times \{-1, 0, 1\}$;
- both at the same time.

We leave out from this part an extension of robot games where the objective is no longer the origin but the set $\mathbb{N} \times 0 \cup \{0\} \times \mathbb{N}$ or $\mathbb{Z} \times 0 \cup \{0\} \times \mathbb{Z}$, because many of the most basic properties of robot games do not hold for this extension, which would hence not bring results that can be used for standard robot games. Still, the tool handles the extension.

For all models that we consider, we need to remember that Eve "can only win when she can win in one round", which is the summary of Proposition 26 on page 60. This is another formulation of the fact that the computation of the attractor stops at the first step if $\mathrm{Pre}(\{(0,0)\}) = \{(0,0)\}$. Geometrically, $W_E \neq \{(0,0)\}$ if, and only if, there is a subset of E that is obtained from A by the reflection across a point $x \neq (0,0)$ (in this case, we have $-2x \in W_E$).

For an extension in dimension two of the arithmetics that we used in dimension one to write Algorithm 3, one can have a look at the articles [PRS05, ST03]. Unfortunately, obtaining a general algorithm for dimension two does not seem feasible with this method, although the shape, like a beam of light, of linear sets in dimension two seen in [PRS05, Fig. 1, page 3] and [ST03, Fig. 2, page 8], which also appears in our figures of the next pages, gave the intuition that the description of Eve's winning set was easy to compute, for example by stepwise computation of sets generated by a greater and greater basis. In dimension two, the problem is to know when to stop the computation: in dimension one, we used results about the Frobenius problem, but this problem does not really exist in higher dimensions. Worse: Eve's winning set is not necessarily a linear set, because a linear set must be generated by a finite basis, which is not guaranteed for two-dimensional robot games,

[1] available at `http://www.lsv.ens-cachan.fr/~reichert/Graphical_tool_robot_games.zip`

and we give a counterexample in Figure 4.3.

4.1 Only horizontal or vertical moves

When Adam and Eve have all their moves in $(\{0\} \times \mathbb{Z}) \cup (\mathbb{Z} \times \{0\})$, we distinguish four cases.

- Either A is a singleton, so we can consider the robot game as a one-player game, which is easy to solve.
- or $A = \{(x, 0), (0, y)\}$, with $x, y \neq 0$, in other words Adam has one move along each axis;
- or $A \subseteq \mathbb{Z} \times \{0\}$ (the case $A \subseteq \{0\} \times \mathbb{Z}$ would be similar), in other words Adam has moves along only one axis;
- or, in all other cases, A has at least two elements along one axis and one along the other. This implies that $W_E = \{(0, 0)\}$.

4.1.1 Adam has one move along each axis

Let us denote $(x, 0)$ and $(0, y)$ for Adam's moves. According to the argument about winning in one round stated in Proposition 26, and because Eve's moves are also only horizontal or vertical, Eve's winning set is not restricted to $\{(0, 0)\}$ if, and only if, she owns at least $(x, 0)$ and $(0, y)$. In this case, $(-x, -y)$ is winning for Eve and, as we can see in Figure 4.1, for the game where $A = \{(-1, 0), (0, 1)\}$ and $E = \{(-1, 0), (4, 0), (-6, 0), (0, 1), (0, 3)\}$, winning counter vectors for Eve may form a set with nontrivial shapes.

4.1.2 Adam has moves along only one axis

When Adam has moves along only one axis, a first idea is to split any play in two steps: first, let Eve reach the axis along which Adam has his moves, then solve a robot game in dimension one. Actually, this does not work, because Eve has at every step to consider Adam's last move to decide when and where to land on the axis. Hence, even if we restrict to strategies where Eve uses for example only vertical moves until the horizontal axis is reached and only horizontal moves afterwards, we cannot say that there exists an integer n such that for all sequences of n moves of Adam, there exists a sequence of n moves of Eve such that a winning counter vector for Eve is reached on the axis along which Adam has his moves. There are in fact n existential and n universal quantifiers that must alternate. We then cannot derive, as hoped, an EXPTIME algorithm according to the idea of splitting plays.

Figure 4.2 gives an exemple of robot game under this restriction: Adam owns the set $A =$

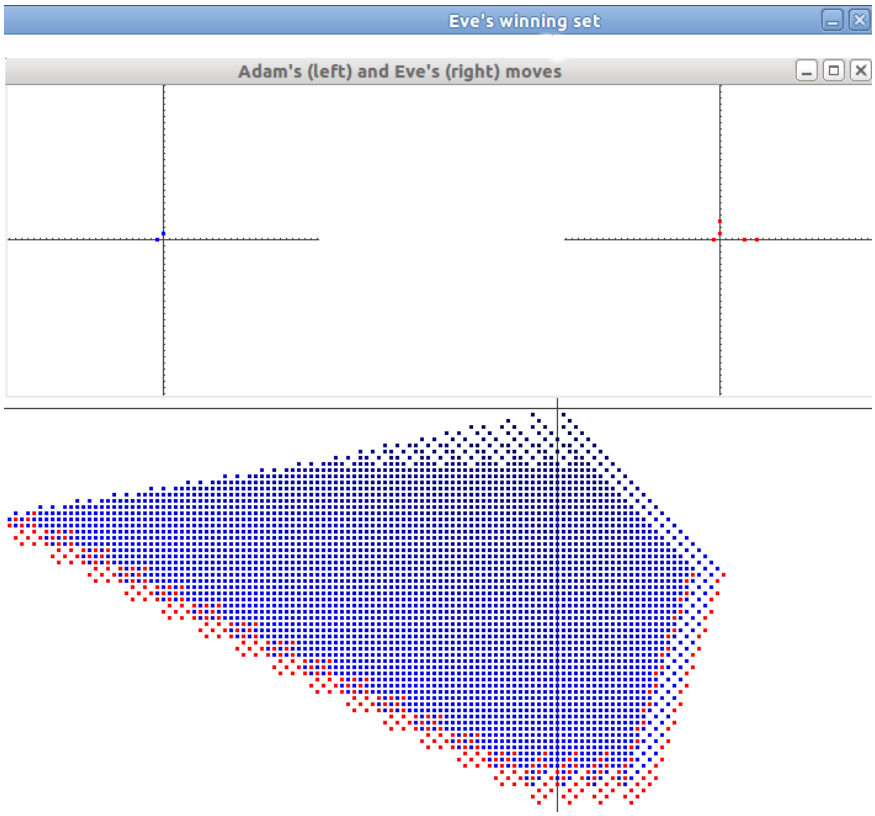

Figure 4.1 – Graphical tool run on the file "test_restr_2_6.txt". Adam has one move along each axis: $(-1,0)$ and $(0,1)$. Eve has the moves $(-1,0)$, $(4,0)$, $(6,0)$, $(0,1)$ and $(0,3)$. Eve's winning set becomes regular below $y=-10$, the above shape is nontrivial, though.

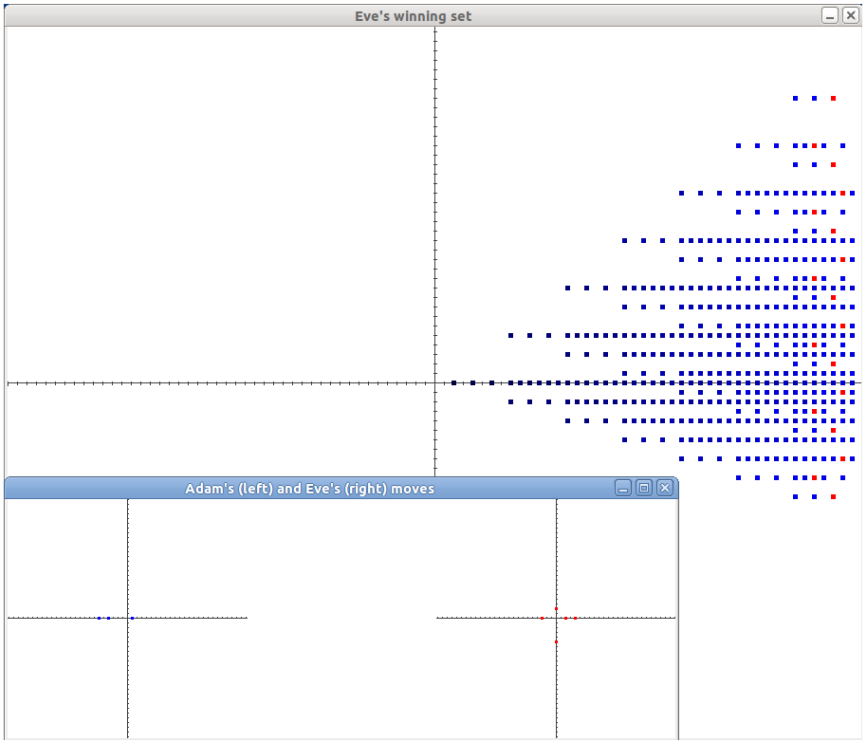

Figure 4.2 – Graphical tool run on the file "test_restr_1.txt". Adam has moves along only one axis : $(-6,0)$, $(-4,0)$ and $(1,0)$. Eve has the moves $(-3,0)$, $(2,0)$, $(4,0)$, $(0,-5)$ and $(0,2)$. Eve's winning set has the same shape on all horizontal lines, only the starting position, i.e., the leftmost position, differs and is given by the formula $8a+6b$, where the ordinate of the line is an integer y such that $y = 5a - 2b$ for the smallest possible nonnegative integers a and b.

$\{(-1,0),(0,1)\}$ and Eve owns the set $\{(-1,0),(4,0),(-6,0),(0,1),(0,3)\}$. The shape of Eve's winning set W_E is regular, although a description of W_E, which we found graphically, is hard to obtain from A and E. This reduces the hope to find an algorithm for determining Eve's winning set in dimension two.

4.2 Unary moves

Everything seems easier when additions are restricted to ± 1 and 0. It is, for example, the framework of the whole articles [BJK10] and [Cha10] about games on VASS. There are exactly 511×511 instances of robot games in dimension two with unary moves, hence there is a constant-time algorithm that looks up the solution for each of them. The goal of this restriction, though, is to search a technique that could be extended to the general case.

According to Proposition 25 on page 59, every non-zero move of Adam defines a region of $\mathbb{Z} \times \mathbb{Z}$ where Eve cannot win. Indeed, if for example Adam owns the move $(1,0)$, then the greatest component along dimension one of a vector in A is 1, and it is greater than or equal to the opposite of the lowest component along dimension one of a vector in E, because we restrict to unary moves. Accordingly, each counter vector with a positive component along dimension one is losing. As a consequence, each horizontal or vertical move of Adam defines a half-plane where Eve cannot win, and each diagonal move of Adam defines a region, covering three quarters of the plane, where Eve cannot win. Moreover, by just looking at the set of moves for Adam, we directly know that the winning set of Eve is included in a set that has one of the following six shapes:

- The singleton $\{(0,0)\}$ (322 out of 511 sets of moves for Adam).
- The whole plane (when $A = (0,0)$, in which case Eve's winning set W_E is $-\langle E \rangle_{\mathbb{N}}$).
- One of the four half-lines that start at $(0,0)$, say $\mathbb{N} \times \{0\}$ and following an axis (136 out of 511 sets), in which case W_E is fully determined by looking at $\mathrm{Pre}(\{(0,0)\})$:
  - If $\mathrm{Pre}(\{(0,0)\}) = \{(0,0)\}$, then $W_E = \{(0,0)\}$;
  - if $(1,0) \in \mathrm{Pre}(\{(0,0)\})$, then $W_E = \mathbb{N} \times \{0\}$;
  - else $\mathrm{Pre}(\{(0,0)\})$ contains $(2,0)$ and possibly $(0,0)$, in both cases $W_E = 2\mathbb{N} \times \{0\}$.
- One of the two axes, say $\mathbb{Z} \times \{0\}$ (4 out of 511 sets), in which case W_E is also fully determined by looking at $\mathrm{Pre}(\{(0,0)\})$: for all integers x that are not zero, $(x,0)$ is in W_E if, and only if, $(\mathrm{sgn}(x),0)$ is in $\mathrm{Pre}(\{(0,0)\})$.
- One of the four half-planes, say $\mathbb{N} \times \mathbb{Z}$ (8 out of 511 sets, 4 of which being trivial because A is a singleton). The interesting case is then $A = \{(0,0),(-1,0)\}$ and, depending on E, the shape of W_E does not have the properties that we expect. For example, in Figure 4.3, where $A = \{(-1,0),(0,0)\}$ and $E = \{(-1,-1),(0,1),(1,1),(1,0)\}$, we have $W_E = \{(0,0)\} \cup$

$(0,-1) + \langle\{(0,-1),(1,0),(2,1)\}\rangle_{\mathbb{N}}$, which is not a linear set.

- One of the four quarters of the plane, say $\mathbb{N}\times\mathbb{N}$ (40 out of 511 sets). The set A is then included in $\{(0,0),(-1,0),(0,-1),(-1,-1)\}$ and contains $(-1,-1)$ or both $(-1,0)$ and $(0,-1)$. An example is given Figure 4.4, where $A = \{(-1,0),(-1,-1)\}$, $E = \{(-1,0),(-1,-1),(0,-1),(1,0),(1,-1)\}$, and W_E is exactly the cone generated by $\{(0,1),(1,2),(2,1)\}$.

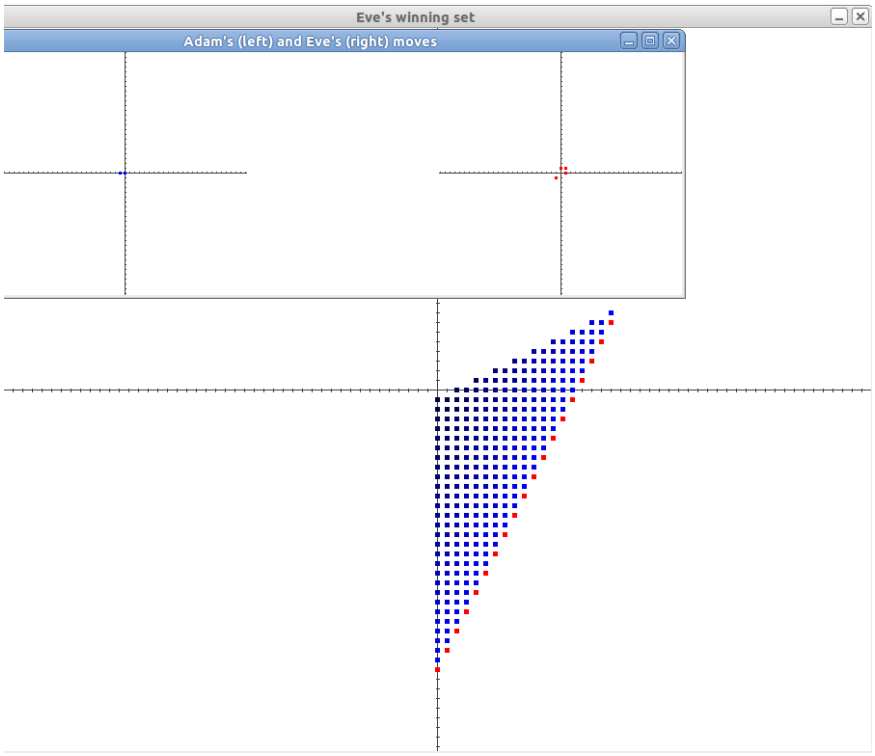

Figure 4.3 – Graphical tool run on the file "test_shrtrg_3.txt". Adam's set of moves is $\{(-1,0),(0,0)\}$, Eve's set of moves is $\{(-1,-1),(0,1),(1,1),(1,0)\}$, and Eve's winning set is $W_E = \{(0,0)\} \cup (0,-1) + \langle\{(0,-1),(1,0),(2,1)\}\rangle_{\mathbb{N}}$, which is not a linear set.

4.3 Unary horizontal and vertical moves, and the "2-2-1 extension"

Restricting to unary moves along one axis means that there are only five possible moves: $(\pm1,0),(0,0)$ and $(0,\pm1)$. There are 31×31 instances of 2RG under this condition, and they can all be solved according to a classification that is similar to the one in the previous section.

First, note that if Adam owns two "opposite" moves, that is, either $(1,0)$ and $(-1,0)$ or $(0,1)$

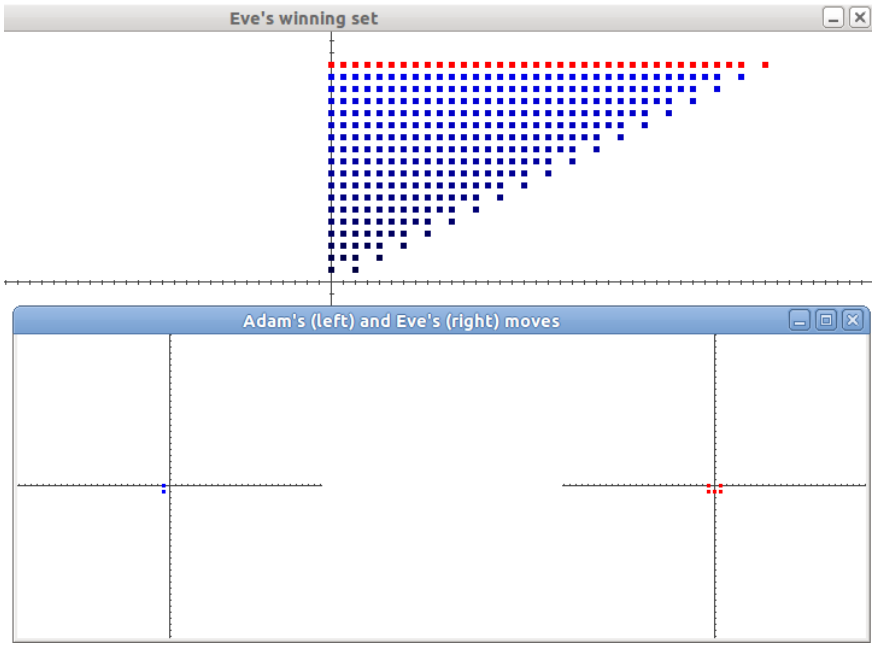

Figure 4.4 – Graphical tool run on the file "test_shrtrg_4.txt". Adam's set of moves is $\{(-1,0),(-1,-1)\}$, Eve's set of moves is $\{(-1,0),(-1,-1),(0,-1),(1,0),(1,-1)\}$, and Eve's winning set is $\langle\{(0,1),(1,2),(2,1)\}\rangle_{\mathbb{N}}$.

and $(0,-1)$, then $W_E = \{(0,0)\}$. By symmetry, we now suppose that $A \subseteq \{(0,0),(-1,0),(0,-1)\}$. We also set aside the easy case of singletons, which lets us with four cases.

- $A = \{(0,0),(-1,0),(0,-1)\}$. Even if Eve owns all five possible moves, we have $W_E = \{(0,0)\}$.
- $A = \{(0,-1),(-1,0)\}$. Then $W_E \neq \{(0,0)\}$ if, and only if, E contains $(0,-1)$ and $(-1,0)$, and in this case, whatever other moves Eve owns, we have $W_E = (1,1)\mathbb{N}$.
- $A = \{(0,0),(-1,0)\}$. Then $W_E \neq \{(0,0)\}$ if, and only if, E contains $(0,0)$ and $(-1,0)$, and in this case, depending on the other moves of Eve, there are four shapes for W_E, as the move $(1,0)$ has no relevance:
  - If neither $(0,-1)$ nor $(0,1)$ are in E, then $W_E = \mathbb{N} \times \{0\}$.
  - If $(0,-1)$ is in E but $(0,1)$ is not, then $W_E = \langle\{(1,0),(0,1)\}\rangle_{\mathbb{N}}$.
  - If $(0,1)$ is in E but $(0,-1)$ is not, then $W_E = \langle\{(1,0),(0,-1)\}\rangle_{\mathbb{N}}$.
  - If $(0,-1)$ and $(0,1)$ are in E, then $W_E = \langle\{(1,0),(0,1),(0,-1)\}\rangle_{\mathbb{N}}$.
- $A = \{(0,0),(0,-1)\}$, it is similar to the previous case.

This result extends well in dimension d: W_E is a hyperpyramid, which is possibly flattened along some axes.

In this section, we call "2-2-1 extension" a more elaborated model, in which A and E are subsets of a fixed set of five moves, among which one is $(0,0)$, two other are on a same line and the two remaining are on another line. For example, a possible set of five moves gathers $(0,0)$, on one line $(2,3)$ and $(6,9)$, and on another line $(1,-2)$ and $(-3,6)$. In the last extension, we mention a case where the fixed set of five possible moves is not the same for Adam and Eve.

4.3.1 First "2-2-1 extension": changing the base

The limit to five possible moves in this subsection seems to be a good framework to find positive results. Let us now make the problem as general as possible, still with the property that Adam and Eve have moves that are included in a set $\{0, \pm x, \pm y\}$ for $x, y \in \mathbb{Z}^2$.

In a first extension, the five possible moves are $\{(0,0), \pm(x_1,y_1), \pm(x_2,y_2)\}$, where (x_1,y_1) and (x_2,y_2) are linearly independent and span, by linear combinations with integer coefficients, a sub-vector space $\mathbb{Y}$ of $\mathbb{Z} \times \mathbb{Z}$.

Because $\{(x_1,y_1),(x_2,y_2)\}$ is a base of $\mathbb{Y}$, there is a linear bijection f from $\mathbb{Y}$ to $\mathbb{Z} \times \mathbb{Z}$ such that $(x_1,y_1) \mapsto (1,0)$ and $(x_2,y_2) \mapsto (0,1)$. If we replace (x_1,y_1) by $(1,0)$ and (x_2,y_2) by $(0,1)$ in a robot game, and we solve the instance that we obtain, we just have to apply f^{-1} to W_E to find

a description of Eve's winning set in the original game.

4.3.2 Second "2-2-1 extension": removing symmetry

According to another restriction, we use horizontal and vertical vectors for the second extension: we remain with five possible moves, but with various sizes: for $a, b, c, d \in \mathbb{Z} \setminus \{0\}$, we use $\{(0,0), (a,0), (b,0), (0,c), (0,d)\}$.

A new case for which it is possible that $W_E \neq \{(0,0)\}$ appears: if both Adam and Eve own $(a,0)$ and $(b,0)$, then the counter vector $(-a-b,0)$ is winning for Eve, and is no longer necessarily $(0,0)$. In general, if Adam owns any set of at least three moves, then $W_E = \{(0,0)\}$. There is only one exception: Adam owns $(a,0)$, $(b,0)$ and $(0,0)$ with $b = 2a$, and Eve owns at least the same moves, in which case W_E is generated by $(-b,0)$, and also by $(-b,-c)$ if $(0,c) \in E$ and by $(-b,-d)$ if $(0,d) \in E$.

Without loss of generality, we have then the following cases:

- $A = \{(a,0), (0,0)\}$. The winning set for Eve can be as hard to describe as in the general case when Adam has only moves along one axis, see Figure 4.5, where $A = \{(-4,0), (0,0)\}$ and $E = \{(-10,0), (-4,0), (0,0), (0,2), (0,-1)\}$.

- $A = \{(a,0), (b,0)\}$. If a and b have the same sign, then $W_E = (-a-b,0)\mathbb{N}$ if E contains $(a,0)$ and $(b,0)$, and $\{(0,0)\}$ else. If a and b have different signs, then the winning set for Eve can there again be like in the previous case, see Figure 4.6, where $A = \{(-4,0), (6,0)\}$ and $E = \{(-4,0), (6,0), (0,0), (0,2), (0,-1)\}$.

- $A = \{(a,0), (0,c)\}$. When for example b is a multiple of a or d is a multiple of c, the winning set for Eve can be as hard to describe as in the general case when Adam has a vertical and an horizontal move, see Figure 4.7, where $A = \{(-1,0), (0,2)\}$ and $E = \{(-2,0), (-1,0), (0,0), (0,2), (0,-1)\}$.

It appears that the difficulties that we raise when we present the case where both players have only horizontal and vertical moves already appear in the restriction where the possible moves are in such a small set as $\{(0,0), (a,0), (b,0), (0,c), (0,d)\}$. The lack of symmetry between the possible moves causes this increase of difficulty to solve robot games.

4.3.3 Third "2-2-1 extension": different sets of possible moves

These observations lead to take another path for the third extension: Two moves along the same direction have the same size, but this size is different for both players. In this section, A is a subset of $\{(\pm a,0), (0,\pm c), (0,0)\}$ and E is a subset of $\{(\pm b,0), (0,\pm d), (0,0)\}$.

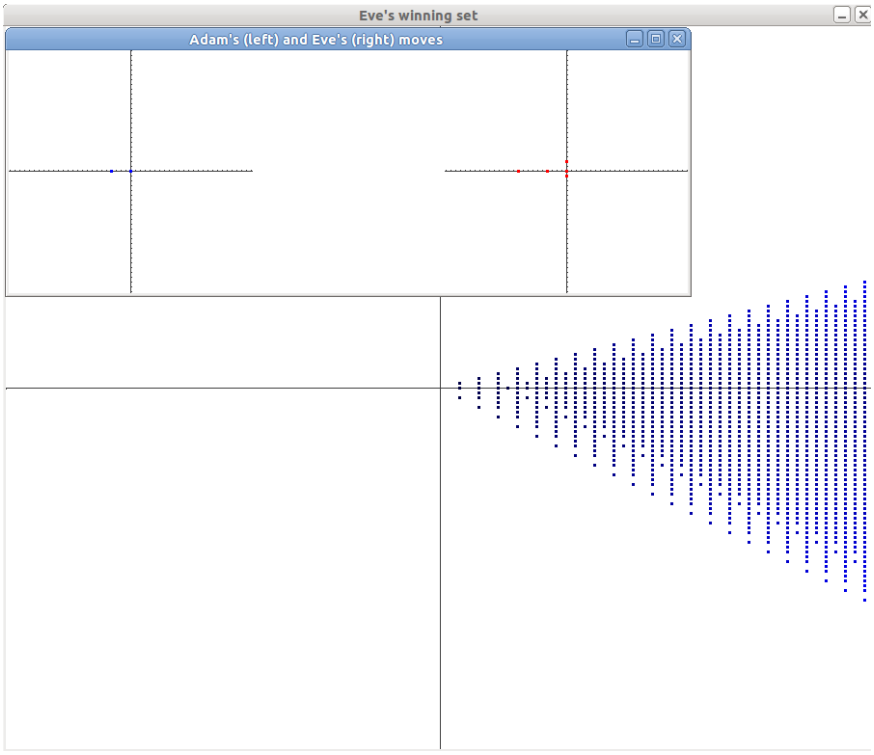

Figure 4.5 – Graphical tool run on the file "test_combi_diff_1.txt". Adam's set of moves is $\{(-4,0),(0,0)\}$, Eve's set of moves is $\{(-10,0),(-4,0),(0,0),(0,2),(0,-1)\}$.

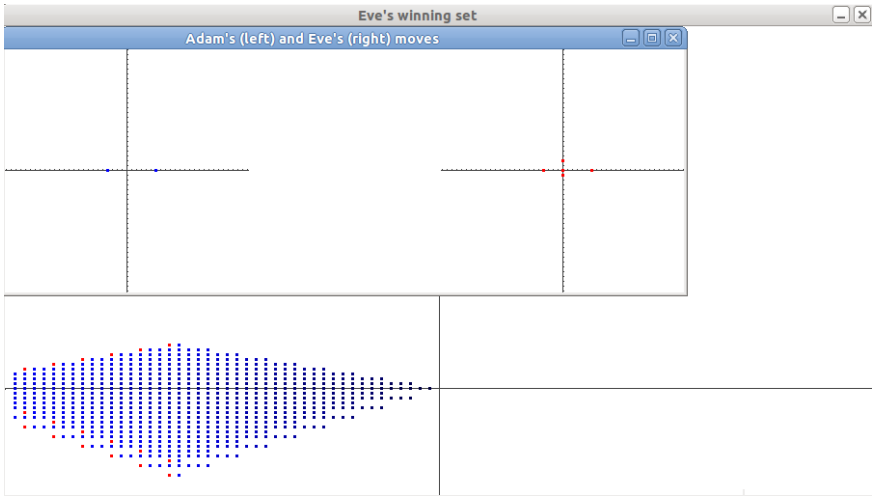

Figure 4.6 – Graphical tool run on the file "test_combi_diff_3.txt". Adam's set of moves is $\{(-4,0),(6,0)\}$, Eve's set of moves is $\{(-4,0),(6,0),(0,0),(0,2),(0,-1)\}$.

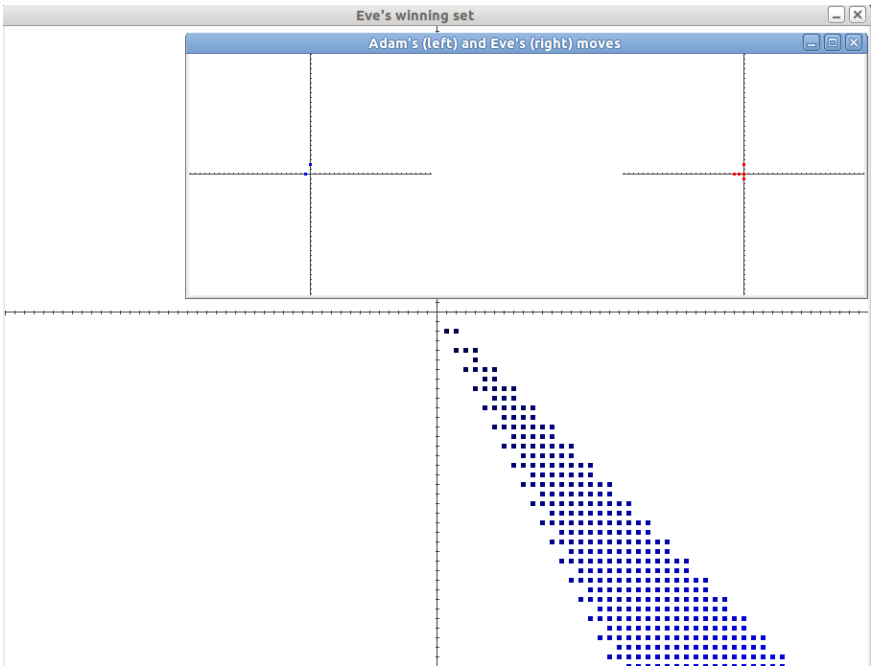

Figure 4.7 – Graphical tool run on the file "test_combi_diff_5.txt". Adam's set of moves is $\{(-1,0),(0,2)\}$, Eve's set of moves is $\{(-2,0),(-1,0),(0,0),(0,2),(0,-1)\}$.

The argument "Eve can only win when she can win in one round" implies that, modulo all symmetries, there are only two possibilities such that neither ($a = \pm b$ & $c = \pm d$) nor $W_E = \{(0,0)\}$ nor A is singleton:

- $A = \{(0,0),(a,0)\}$, $a = 2b$ and Eve owns at least $\pm b$. The shape of W_E depends on the other moves of Eve, apart from $(0,0)$, which has no influence.
  - If neither $(0,d)$ nor $(0,-d)$ are in E, then $W_E = -(b,0)\mathbb{N}$.
  - If $(0,d) \in E$ but not $(0,-d)$, then $W_E = \langle\{(-b,0),(-2b,-d)\}\rangle_{\mathbb{N}}$. Analoguous if $(0,-d) \in E$ but not $(0,d)$.
  - If $(0,\pm d) \in E$, then $W_E = \langle\{(-b,0),(-2b,-d),(-2b,d)\}\rangle_{\mathbb{N}}$.
- $A = \{(\pm a,0)\}$, $b = 2a$ and Eve owns at least $(0,0)$ and one move among $(b,0)$ and $(-b,0)$. Let us denote $\mathbb{Y}$ for the sub-vector space that $(b,0)$ and $(0,d)$ generate. The shape of W_E is easy to describe given E, it can be the intersection of $\mathbb{Y}$ with a half-line, a line, a cone, a half-plane or the full plane.

This result confirms what we noticed in the previous subsection: symmetry is fundamental to have winning sets that are simple to describe. In some sense, moves a and $-a$ do not really act like two different moves, it is rather doing an action and cancelling it.

4.3.4 Fourth "2-2-1 extension": removing parallelism

This time, Adam's moves and Eve's moves do not share all their directions: Adam's set of moves A is a subset of $\{(0,0),\pm v_1,\pm v_2\}$, Eve's set of moves E is a subset of $\{(0,0),\pm v_3,\pm v_4\}$, where we suppose that v_1 or v_2 is neither parallel to v_3 nor to v_4.

Once again because "Eve can only win when she can win in one round", as soon as Adam has at least two moves, a disjunction of equations that link v_1, v_2, v_3 and v_4 must be verified in order that $W_E \neq \{(0,0)\}$.

For example, if $A = \{(0,0),v_1\}$, then we look for two moves in E that have v_1 as vector difference. In other words, $v_1 = \pm v_3 \pm v_4$.

Modulo symmetry, the only cases that do not imply directly $W_E = \{(0,0)\}$ are the following:

- $A = \{(0,0),v_1\}$.
- $A = \{\pm v_1\}$.
- $A = \{(0,0),v_1,v_2\}$ provided that other equations hold.

- $A = \{v_1, v_2\}$ (same).
- $A = \{\pm v_1, v_2\}$ (same).

Figure 4.8 presents an example in another context, where $A = \{(0,0), v_1, v_2\}$ and $E = \{(0,0), \pm v_3, \pm v_4\}$, with $v_1 = (-1,0)$, $v_2 = (1,2)$, $v_3 = v_1$, and $v_4 = v_2 - v_3 = (2,2)$. Here, Eve's winning set is $\langle\{(0,1),(-1,-2)\}\rangle_{\mathbb{N}}$.

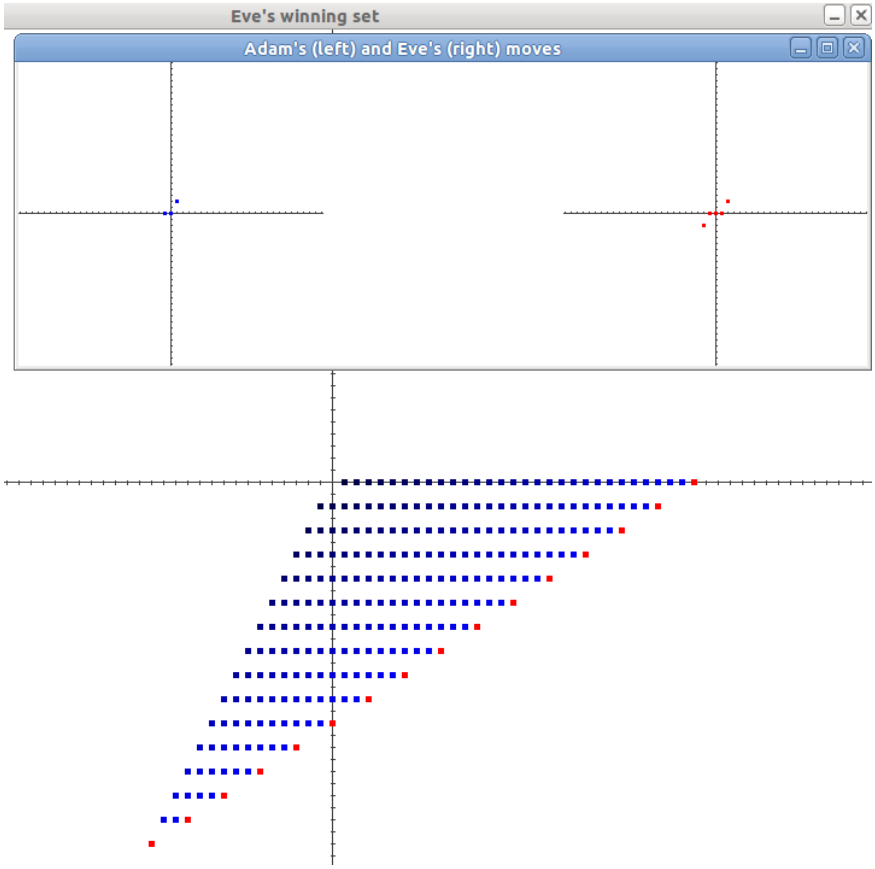

Figure 4.8 – Graphical tool run on the file "test_combi_indep_pire_1.txt". Adam's set of moves is $\{(0,0),(-1,0),(1,2)\}$, Eve's set of moves is $\{(0,0),(-1,0),(1,0),(2,2),(-2,-2)\}$.

As a matter of fact, our study of robot games in dimension two reveals that the models that we consider seem often either trivial or as complicated as the general case. Whether the problem of determining the winner of a two-dimensional robot games is decidable or not is left as open. Actually, even though we have seen that in many cases Eve's winning set has a regular shape, we cannot lend towards believing that the problem is decidable: in dimension three, we have proved indecidability and there are most certainly many examples where Eve's winning set has a regular shape, which are unfortunately much harder to represent on a screen.

Chapter 5

Conclusion

In this thesis, we considered the problem of deciding the winner in one-player or two-player reachability games on systems with one or more counters under various semantics for counter updates. The summary of our results is that decidability seems to require that there is only one counter, or only one player, whichever is the semantics of the system.

The tables of this section recall all decidability and complexity results in the manuscript, with a reference to the section, or in the literature, depending on the model. The symbol $*$ stands for folklore.

Objective	$\mathbb{Z}$ semantics	VASS semantics	non-blocking VASS semantics
$(q, 0)$ (dim. 1)	NP-complete [*,2.2.1]	NP-complete [*]	P [2.3.2]
(q, c) (dim. 1)	NP-complete [*,2.2.1]	NP-complete [*]	NP-complete [2.3]
$(q, 0)$ (dim. 1), unary	P [*]		
$(q, 0^d)$ (dim. d)	NP-complete [*,2.2.1]	decidable [[Kos82]]	not considered

Figure 5.1 – Reachability problem (one-player version)

Determining the winner of a counter reachability game according to the objectives that we use, i.e., the counter vector must be equal to a target vector (usually zero) and the vertex must be the only goal vertex, is undecidable in dimension two under all semantics. Hence, we spare the drawing of a table with "undecidable" everywhere. The case of dimension one, with decidability results, shows more diversity: see the table in Figure 5.2.

Objective	$\mathbb{Z}$ semantics	VASS semantics	NBVASS semantics
(q,0), unary	PSPACE-C [2.2.3]	PSPACE-C [[Ser06],[BJK10]]	P [2.3.2]
(q,1), unary	PSPACE-C [2.2.3]	PSPACE-C [[Ser06],[BJK10]+2.2.3]	PSPACE-C [2.3.1]
(q,0)	EXPSPACE-C [[Hun14]]	EXPSPACE-C [[Hun14]]	NP [2.3.2]
(q,1)	EXPSPACE-C [[Hun14]]	EXPSPACE-C [[Hun14]]	EXPSPACE-C [[Hun14]]

Figure 5.2 – Reachability games in dimension one

The EXPSPACE-hardness in the cells of the bottom-left corner of the table in Figure 5.2 has been proved by Paul Hunter in [Hun14], membership in EXPSPACE is obtained by splitting edges to obtain a unary arena of exponential size.

For the extensions that we study, our results do not depend on the semantics.

Model	Objective	Decidability/complexity
Two players, with resets	(q,c), c fixed, unary	PSPACE-complete [2.4]
Two players, with resets	(q,c), c fixed	EXPSPACE-C [2.4,[Hun14]]
One player, with affine updates	(q,v), v fixed, dim. ≥ 2	Undecidable [2.5]
One player, with hierarchical updates	(q,v), v fixed	NP [2.6.1]

Figure 5.3 – Reachability games, variants

Finally, for robot games, our main results are EXPTIME-completeness in dimension one and undecidability in dimension three. For dimension two, the problem is open in the general case, but we have some refinements. In the next table, "restricted choice" means that Eve's set of moves is a subset of $\{0, \pm v_1, \pm v_2\}$ and Adam's set of moves is a subset of $\{0, \pm v_3, \pm v_4\}$, for given vectors v_1, v_2, v_3 and v_4. In fact, more than being solvable in polynomial time, the decision problem often has an immediate solution. For example, when $E = \{v_1, v_2\}$ and $A = \{v_3, v_4\}$, Eve's winning set is $\{0^d\}$ if, and only if, $v_1 - v_2 \neq \pm(v_3 - v_4)$. On the other hand, if $v_1 - v_2 = v_3 - v_4$, then Eve's winning set is $(-v_1 - v_4)\mathbb{N} = (-v_2 - v_3)\mathbb{N}$. This property has been used in [HNP15], where the authors introduce the notion of degree of a robot game, i.e., the maximal number of moves of Adam and Eve.

Dimension	Particular case	Decidability/complexity
1		EXPTIME-complete [3.1.4,3.2]
2	Eve has states	Undecidable [3.3.2]
3		Undecidable [3.3.4]
d	One player	NP-complete [inherited from 2.2.1]

Figure 5.4 – Robot games

Further directions may include extensions of the models that we consider to games on counters with letters on the labels of edges. Recently, the authors of [HHNP15] have made a link between a problem that they prove to be undecidable on weighted automata and reachability games. This raises the issue of the effects of adding an alphabet to our games, as well as the possibility of describing accepting runs of counter systems with languages.

Bibliography

[ABD08] P. A. Abdulla, A. Bouajjani, and J. D'Orso. Monotonic and downward closed games. *Journal of Logic and Computation*, 18(1):153–169, 2008.

[ABH08] M. F. Atig, B. Bollig, and P. Habermehl. Emptiness of multi-pushdown automata is 2etime-complete. In *Proceedings of the 12th International Conference on Developments in Language Theory (DLT)*, pages 121–133, 2008.

[AM09] P. A. Abdulla and R. Mayr. Minimal cost reachability/coverability in priced timed petri nets. In *Proceedings of the 12th International Conference on Foundations of Software Science and Computational Structures (FOSSACS)*, pages 348–363, 2009.

[AR13] A. Arul and J. Reichert. The complexity of robot games on the integer line. In *Proceedings of the 11th Workshop on Quantitative Aspects of Programming Languages and Systems (QAPL)*, volume 117 of *EPTCS*, pages 132–146, 2013.

[Ati10] M. F. Atig. Global model checking of ordered multi-pushdown systems. In *Proceedings of the IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 216–227, 2010.

[BCCCR96] L. Breveglieri, A. Cherubini, C. Citrini, and S. Crespi-Reghizzi. Multi-push-down languages and grammars. *International Journal of Foundations of Computer Science*, 7(3):253–292, 1996.

[BCKN12] T. Brázdil, A. Chatterjee, A. Kucera, and P. Novotný. Efficient controller synthesis for consumption games with multiple resource types. In *Proceedings of the 24th International Conference on Computer Aided Verification (CAV)*, pages 23–38, 2012.

[BFG+14] M. Blondin, A. Finkel, S. Göller, C. Haase, and P. McKenzie. Reachability in two-dimensional vector addition systems with states is pspace-complete. *CoRR*, abs/1412.4259, 2014.

[BFL+08] P. Bouyer, U. Fahrenberg, K. G. Larsen, N. Markey, and J. Srba. Infinite runs in weighted timed automata with energy constraints. In *Proceedings of the 6th Interna-*

tional Conference on Formal Modelling and Analysis of Timed Systems (FORMATS), pages 33–47, 2008.

[BFLZ10] R. Bonnet, A. Finkel, J. Leroux, and M. Zeitoun. Place-boundedness for vector addition systems with one zero-test. pages 192–203, 2010.

[BFLZ12] R. Bonnet, A. Finkel, J. Leroux, and M. Zeitoun. Model checking vector addition systems with one zero-test. *Logical Methods in Computer Science*, 8(2), 2012.

[BHSS12] B. Bérard, S. Haddad, M. Sassolas, and N. Sznajder. Concurrent games on VASS with inhibition. In *Proceedings of the 23rd International Conference on Concurrency Theory (CONCUR)*, pages 39–52, 2012.

[BJK10] T. Brázdil, P. Jancar, and A. Kucera. Reachability games on extended vector addition systems with states. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP) - Part II*, pages 478–489, 2010.

[BM99] A. Bouajjani and R. Mayr. Model checking lossy vector addition systems. In *Proceedings of the 16th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 323–333, 1999.

[Bon11] R. Bonnet. Decidability of LTL for vector addition systems with one zero-test. In *Proceedings of the 5th International Workshop on Reachability Problems (RP)*, pages 85–95, 2011.

[CD10] K. Chatterjee and L. Doyen. Energy parity games. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP) - Part II*, pages 599–610, 2010.

[CD11] K. Chatterjee and L. Doyen. Energy and mean-payoff parity markov decision processes. In *Proceedings of the 36th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 206–218, 2011.

[CDHR10] K. Chatterjee, L. Doyen, T. A. Henzinger, and J.-F. Raskin. Generalized mean-payoff and energy games. *CoRR*, abs/1007.1669, 2010.

[Cha10] J. Chaloupka. Z-reachability problem for games on 2-dimensional vector addition systems with states is in p. In *Proceedings of the 4th International Workshop on Reachability Problems (RP)*, pages 104–119, 2010.

[CLRS09] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms (third edition)*. MIT Press, 2009.

[DFS98] C. Dufourd, A. Finkel, and P. Schnoebelen. Reset nets between decidability and undecidability. In *Proceedings of the 25th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 103–115, 1998.

[DR13] L. Doyen and A. Rabinovich. Robot games. Research Report LSV-13-02, Laboratoire Spécification et Vérification, ENS Cachan, France, January 2013. 2 pages.

[Eps12] R. A. Epstein. *The theory of gambling and statistical logic*, page 351. Academic Press, 2012.

[FGH13] A. Finkel, S. Göller, and S. Haase. Reachability in register machines with polynomial updates. In *Proceedings of the 38th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 409–420, 2013.

[FJLS11] U. Fahrenberg, L. Juhl, K. G. Larsen, and J. Srba. Energy games in multiweighted automata. In *Proceedings of the 8th International Colloquium on Theoretical Aspects of Computing (ICTAC)*, pages 95–115, 2011.

[FS10] A. Finkel and A. Sangnier. Mixing coverability and reachability to analyze VASS with one zero-test. In *Proceedings of the 36th Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, pages 394–406, 2010.

[GS53] D. Gale and F. M. Stewart. Infinite games with perfect information. In *Contributions to the theory of games, vol. 2*, Annals of Mathematics Studies, no. 28, pages 245–266. Princeton University Press, Princeton, N. J., 1953.

[GTW02] E. Grädel, W. Thomas, and T. Wilke, editors. *Automata, logics, and infinite games: a guide to current research*, volume 2500. Springer Science & Business Media, 2002.

[HH14] C. Haase and S. Halfon. Integer vector addition systems. *arXiv preprint arXiv:1406.2590*, 2014.

[HHNP15] V. Halava, T. Harju, R. Niskanen, and I. Potapov. Weighted automata on infinite words in the context of attacker-defender games. In *Proceedings of the 11th Conference on Computability in Europe (CiE) - Evolving Computability*, pages 206–215, 2015.

[HKOW09] C. Haase, S. Kreutzer, J. Ouaknine, and J. Worrell. Reachability in succinct and parametric one-counter automata. In *Proceedings of the 20th International Conference on Concurrency Theory (CONCUR)*, pages 369–383, 2009.

[HNP15] V. Halava, R. Niskanen, and I. Potapov. On robot games of degree two. In *Proceedings of the 9th International Conference on Language and Automata Theory and Applications (LATA)*, pages 224–236, 2015.

[HP79] J. E. Hopcroft and J.-J. Pansiot. On the reachability problem for 5-dimensional vector addition systems. *Theoretical Computer Science*, 8:135–159, 1979.

[HRHY86] R. R. Howell, L. E. Rosier, D. T. Huynh, and H.-C. Yen. Some complexity bounds for problems concerning finite and 2-dimensional vector addition systems with states. *Theoretical Computer Science*, 46(0):107 – 140, 1986.

[Hun14] P. Hunter. Reachability in transition graphs of alternating one-counter machines. *ArXiv e-prints*, jul 2014.

[JLS07] M. Jurdzinski, F. Laroussinie, and J. Sproston. Model checking probabilistic timed automata with one or two clocks. In *Proceedings of the 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pages 170–184, 2007.

[KM69] R. M. Karp and R. E. Miller. Parallel program schemata. *The Journal of Computer and System Sciences*, 3(2):147–195, 1969.

[Kos82] S. R. Kosaraju. Decidability of reachability in vector addition systems (preliminary version). In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC '82, pages 267–281, New York, NY, USA, 1982. ACM.

[KS88] S. R. Kosaraju and G. F. Sullivan. Detecting cycles in dynamic graphs in polynomial time. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, pages 398–406, 1988.

[May84] E. W. Mayr. An algorithm for the general petri net reachability problem. *Society for Industrial and Applied Mathematics (SIAM) Journal of Computing*, 13(3):441–460, 1984.

[Min67] M. L. Minsky. *Computation*. Prentice-Hall, 1967.

[MS72] A. R. Meyer and L. J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Proceedings of the 13th Annual Symposium on Switching and Automata Theory (SWAT)*, pages 125–129, 1972.

[Pet62] C. A. Petri. Fundamentals of a theory of asynchronous information flow. In *Proceedings of the International Federation for Information Processing (IFIP) Congress*, pages 386–390, 1962.

[Pos46] E. L. Post. A variant of a recursively unsolvable problem. *Bulletin of the American Mathematical Society*, 52(4):264–268, 1946.

[PRS05] P. A. B. Pleasants, H. Ray, and J. Simpson. The Frobenius problem on lattices. *Australasian Journal of Combinatorics*, 2005.

[PS82] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[Rei95] K. Reinhardt. Reachability in petri nets with inhibitor arcs. `http://www-ti.informatik.uni-tuebingen.de/~reinhard/reapet.dvi`, unpublished, 1995.

[Rei08] K. Reinhardt. Reachability in petri nets with inhibitor arcs. *Electronic Notes in*

Theoretical Computer Science, 223:239–264, 2008.

[Rei13] J. Reichert. On the complexity of counter reachability games. In *Proceedings of the 7th International Workshop on Reachability Problems (RP)*, pages 196–208, 2013.

[Ser06] O. Serre. Parity games played on transition graphs of one-counter processes. In *Proceedings of the 9th International Conference on Foundations of Software Science and Computation Structures (FOSSACS)*, pages 337–351, 2006.

[Set09] A. Seth. Games on multi-stack pushdown systems. In *Proceedings of the Symposium on Logical Foundations of Computer Science (LFCS)*, pages 395–408, 2009.

[Set10] A. Seth. Global reachability in bounded phase multi-stack pushdown systems. In *Proceedings of the 22nd International Conference on Computer Aided Verification (CAV)*, pages 615–628, 2010.

[ST03] R. J. Simpson and R. Tijdeman. Multi-dimensional versions of a theorem of fine and wilf and a formula of sylvester. volume 131, pages 1661–1671, 2003.

[Syl82] J. J. Sylvester. On subvariants, i.e. semi-invariants to binary quantics of an unlimited order. *American Journal of Mathematics*, 5(1):pp. 79–136, 1882.

[Wil78] H. S. Wilf. A circle-of-lights algorithm for the "money-changing problem". *The American Mathematical Monthly*, 85(7):pp. 562–565, 1978.

Index

Index of figures

Chapter 2

Introduction

Reachability games on counter systems under the $\mathbb{Z}$ semantics

Reachability games on counter systems under the Non-blocking VASS semantics in dimension one

Reachability games on reset counter systems

Hierarchical counter systems

Chapter 3

Introduction

Lower complexity bound in dimension one

Undecidability in dimension three

Chapter 4