Mortality of Iterated Piecewise Affine Functions over the Integers: Decidability and Complexity

Amir M. Ben-Amram

The Academic College of Tel-Aviv Yaffo
amirben@mta.ac.il

Abstract

In the theory of discrete-time dynamical systems, one studies the limiting behaviour of processes defined by iterating a fixed function f over a given space. A much-studied case involves piecewise affine functions on $\mathbb{R}^n$. Blondel et al. (2001) studied the decidability of questions such as *mortality* for such functions with rational coefficients. *Mortality* means that every trajectory includes a 0; if the iteration is seen as a loop `while` $(x \neq 0)$ x `:=` $f(x)$, mortality means that the loop is guaranteed to terminate.

Blondel et al. proved that the problems are undecidable when the dimension n of the state space is at least two. They assume that the variables range over the rationals; this is an essential assumption. From a program analysis (and discrete Computability) viewpoint, it would be more interesting to consider integer-valued variables.

This paper establishes (un)decidability results for the *integer* setting. We show that also over integers, undecidability (moreover, Π^0_2 completeness) begins at two dimensions. We further investigate the effect of several restrictions on the iterated functions. Specifically, we consider bounding the size of the partition defining f, and restricting the coefficients of the linear components. In the decidable cases, we give complexity results. The complexity is PTIME for affine functions, but for piecewise-affine ones it is PSPACE-complete. The undecidability proofs use some variants of the Collatz problem, which may be of independent interest.

1998 ACM Subject Classification F.1.0 Computation by abstract devices – General

Keywords and phrases discrete-time dynamical systems, termination, Collatz problem

Digital Object Identifier 10.4230/LIPIcs.STACS.2013.514

1 Introduction

The purpose of this paper is to study some computational problems regarding the asymptotic behaviour of discrete-time dynamical systems, in particular problems related to questions about the termination of simple programs. In the context of this paper, an (n-dimensional) *discrete-time dynamical system* is defined by $x_{t+1} = f(x_t)$, where $f : \mathbb{R}^n \to \mathbb{R}^n$. For a given initial point x_0, the sequence so generated is called *a trajectory.* A class of functions much studied in the Dynamical System literature is piecewise affine functions (Definition 3 at the end of this section). Some of the central problems regarding such systems involve their asymptotic behavior, among which are *global convergence* and *mortality*, defined next.

▶ **Definition 1.** Let f be an arbitrary map on $\mathbb{R}^n$; let 0 be the origin. We call f *globally convergent to zero* if for every initial point x_0, the trajectory $x_{t+1} = f(x_t)$ converges to 0 (the words "to zero" and sometimes "globally" may be omitted in the sequel). We call f *mortal* if for every initial point, the trajectory reaches the origin.

These problems were studied from a Computability viewpoint by Blondel et al. [3]. They considered piecewise affine functions f, where the coefficients are all rational (this is important since we are considering computability in the traditional, discrete sense).

30th Symposium on Theoretical Aspects of Computer Science (STACS'13).
Editors: Natacha Portier and Thomas Wilke; pp. 514–525
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
SYMPOSIUM ON THEORETICAL ASPECTS OF COMPUTER SCIENCE

▶ **Theorem 2.** [3] . *The following problems are undecidable for all* $n \geq 2$*: Given a piecewise affine function* $f : \mathbb{R}^n \to \mathbb{R}^n$ *(with rational coefficients), is it globally convergent? Is it mortal?*

Global convergence is decidable for $n = 1$ *when the function is continuous.*

Among the many decision problems studied for dynamical systems, mortality is the most appealing in its discrete setting since it is a restricted halting problem for a very simple type of program (more on the connection to program termination problems below). The global convergence problem as defined above is very closely related (in a discrete setting, a sequence x_t converges to 0 if and only if it is eventually constantly zero) and we will treat both problems throughout the following sections.

The main contribution of this paper is to establish for the integer setting results similar to Theorem 2. The proofs in [3] do not apply to this setting, since they are based on encoding the state of a computation (say, a Turing-machine tape) in the fractional digits of a number; unlimited precision is essential. To handle a discrete (or limited precision) setting, different proof techniques are necessary. As in the continuous case, we have decidability for one dimension and undecidability for two or more. The new undecidability result is stronger than the one in [3] in that it is achieved for a class of functions where there is a fixed bound on the number of regions in the partition defining f ([3] left this case open; but one can strengthen their result in this way, too). Furthermore, we prove Π^0_2 completeness, which is a sharper result than mere undecidability.

Next, we consider some other restrictions on the space of functions. Section 4 shows undecidability for functions of a very simple form, $f(x) = (x_{i_1} + b_1, x_{i_2} + b_2)$.

We should point out that when there is only one region (that is, f is affine) all problems considered in this paper are decidable in any dimension, and moreover in polynomial time; this follows from Linear-Algebraic techniques as used for linear discrete dynamical systems over the reals. However, even when we have one convex region in which the function is defined as an affine function, and is zero elsewhere, its analysis becomes challenging and, in fact, is still an open problem [5].

Section 5) considers the one-dimensional case, where the problems are decidable, and we concentrate on complexity results. While the affine case is PTIME, we show the one-dimensional piecewise-affine case to be PSPACE-complete.

In Section 6 we mention a couple of similar problems whose solutions follow easily, whereas the conclusion (Sect. 7) presents a selection of open problems.

To conclude this introduction, here are a few comments on the background to this research. The connection of Dynamical System Theory to the Theory of Computation is obvious—any traditional model of computation is a discrete-time dynamical system. However, most literature in Dynamical System Theory refers to a continuous state-space, whereas in the Theory of Computation, most models are discrete (but analog models *are* studied and cross-fertilization with Dynamical System Theory is evident; see for example [19]).

Taking classical (discrete) computability to continuous dynamics, Moore [17] discusses the significance of undecidability (in the Turing sense) to dynamical systems. He shows that a TM can be simulated by a piecewise-affine map on the plane—the method is quite similar to the one used by Blondel et al. He concludes that for such a map, the set of points on which the sequence converges (to a particular zone) is not recursive. Koiran, Cosnard, and Garzon showed that such simulations can be done already in two dimensions, but not in one [13]. Blondel and Tsiklitis [4] survey applications of discrete computability and complexity to dynamical systems, including the above-cited results.

The author's interest in the mortality problem and its variants is due to their interpretation as special cases of the halting problem, a.k.a. program termination (the latter term often refers to termination for any input, that is, a global property as those studied here). Decision procedures for the termination of *simple loops*, where a fixed (loop-free) computation is iterated until an end-condition is met, have gained much interest in program analysis and several heuristic approaches have been proposed (e.g., linear ranking functions [8, 18, 1]). These works concentrate on integer data, and on functions which are linear, piecewise linear, or defined by linear constraints (which is a wider class). In [20], Tiwari draws on inspiration from Dynamical System Theory to solve a termination problem for loops with an affine-linear update function—however, over the reals. Consequently, Braverman [5] tackled the problem for the rationals and integers. Passing from the real-number world to the integers is sometimes quite a challenge as the theory of integers has many surprises of its own. A notorious example is the solution of multivariate polynomial equations—or, more generally, quantifier elimination—decidable for the reals, but undecidable for integers. Another classic example of an integer-specific problem is the Collatz problem (or "$3x + 1$ problem") [15]. Lagarias' excellent volume shows clearly that this problem is related both to dynamical system theory and to computability theory. Regarding the latter, Conway [9, 10] and subsequent works [12, 7, 14] proved undecidability results for certain *generalized Collatz problems* by showing how to simulate a counter machine. We shall make essential use of this idea, specifically of the approach of [14], who were first to show Π^0_2 hardness of mortality for a generalized Collatz function.

For completeness, we should mention that there are kinds of dynamical systems very remote from our subject, the interested reader is referred to [6].

Preliminary definitions

A closed (respectively open) *half-space* of $\mathbb{R}^n$ is the set defined by $\{x \in \mathbb{R}^n : cx + d \geq 0\}$ (respectively > 0) where $c \in \mathbb{R}^n, d \in \mathbb{R}$. We are interested in *rational* half-spaces, where the components of c, d are rational. A (rational) *convex polyhedron* is the intersection of a finite number of (closed or open) half-spaces, which we sometimes call *the constraints*.

▶ **Definition 3.** A *piecewise affine function* on $\mathbb{R}^n$ (respectively $\mathbb{Z}^n$) is a function f where

$$f(x) = A_i x + b_i \quad \text{for } x \in H_i \text{ (respectively, } H_i \cap \mathbb{Z}^n) \tag{1}$$

where the sets $H_1, \dots, H_p$ are an exhaustive partition of $\mathbb{R}^n$ into p convex rational polyhedra, and for $i = 1, \dots, p$, $A_i \in \mathbb{Q}^{n\times n}$ and $b_i \in \mathbb{Q}^n$ (respectively, $\mathbb{Z}^{n\times n}$ and $\mathbb{Z}^n$).

The restriction to polyhedra which are convex is somewhat arbitrary, but follows the definition in [3] and other related publications. Section 3 discusses an implication of this restriction.

We use the notation $[a, b]$ for an interval of integers, namely $\{a, a + 1, \dots, b\}$.

The counter machine model, due to Minsky [16], is well known. The details of the definition vary in the literature, but the differences are not essential. At this point, it should suffice to remark that we write a state of the machine as $(i, \langle r_1, \dots, r_n\rangle)$ where i is the internal state and r_j the contents of register (counter) R_j.

Π^0_2 is the class of decision problems that can be expressed by a formula of the form $(\forall z)(\exists y)P(x, y, z)$ with P recursive. This class properly contains RE (characterized by formulas $(\exists y)P(x, y)$). A standard Π^0_2-complete set is the totality problem (termination on all inputs) for Turing machines, as well as counter machines.

2 Undecidability in Two Dimensions

Blondel et al. prove their undecidability results for Generalized Collatz Problems by reducing from the *mortality problem for counter machines* (the set of CMs that halt on every given configuration). Indeed, the PAF mortality problem is similar to the GCP, which is also a problem of reachability (of 1 instead of 0), but the functions considered in the GCP are not piecewise affine; their expression involves division and remainders, which make it easier to encode computations and simulate counter machines. Since the reductions in [3] are based on fractional numbers, we introduce new reductions. In addition, we rely on some proof techniques from [14], who proved that mortality of counter machines is Π^0_2-complete (leading to a similar result for GCPs). Note that hardness of mortality does not follow easily from hardness of totality, since many programs, while halting from all initial states, still diverge if started at a configuration that is not reachable in a proper computation (i.e., from an initial state).

▶ **Definition 4.** A function $g : \mathbb{N}_+ \to \mathbb{N}_+$ is called a *generalized Collatz function* if there is an integer $m > 0$, positive integers $\{a_0, \ldots, a_{m-1}\}$ and non-negative integers $\{b_0, \ldots, b_{m-1}\}$, such that whenever $x \equiv i \bmod m$, $g(x) = a_i(x - i)/m + b_i$.

A *standard representation* of g is the list $m, a_0, b_0, \ldots, a_{m-1}, b_{m-1}$.

The standard Collatz function is usually described by $g(x) = 3x + 1$ if x is odd, $g(x) = x/2$ if x is even. In our notation, it is given by $m = 2$, $a_0 = 1, b_0 = 0$, $a_1 = 6, b_1 = 4$.

▶ **Definition 5.** GCP (for Generalized Collatz Problem) is the problem of deciding, from a standard representation of g, whether every trajectory of g reaches 1.

▶ **Theorem 6.** [14] *GCP is Π^0_2-complete.*

Our first result is

▶ **Theorem 7.** *Global convergence and mortality over $\mathbb{Z}$ of piecewise affine functions with integer coefficients is a Π^0_2-complete problem.*

Proof. Like the GCP, our problem is clearly a "∀∃" problem, hence belonging to Π^0_2. For Π^0_2-hardness, we reduce from the GCP. Given a description $\langle m, a_0, b_0, \ldots, a_{m-1}, b_{m-1}\rangle$ of a generalized Collatz function g, our reduction produces the function f defined by the $m + 1$ cases Table 1, where for convenience every region has a label.

region label	constraints	$f(x, y)$
D	$x>m,\ y \ge 0$	$(x - m, y + 1)$
R_0	$x=0\ ,\ y > 0$	$(a_0 y + b_0, 0)$
R_1	$x=1\ ,\ y > 0$	$(a_1 y + b_1, 0)$
R_2	$x=2\ ,\ y \ge 0$	$(a_2 y + b_2, 0)$
$\ldots$		
R_{m-1}	$x=m,\ y \ge 0$	$(a_m y + b_m, 0)$
Z	elsewhere	$(0, 0)$

Table 1 PAF representing a GCP.

To simulate a Collatz sequence generated by g, we repeatedly apply f starting from $(x_0, 0)$. Observe that started at $(x, 0)$ for $x > 1$, the computation will stay in Region D (the *division* region) until obtaining the result $(i, (x-i)/m)$ where $x \equiv i \bmod m$. Computation will then reach one of Regions R_0 through R_{m-1} and apply the appropriate case of g, producing $(g(x), 0)$. This process iterates until arriving at $(1, 0)$, which indicates the convergence of the Collatz sequence and is mapped to $(0, 0)$, the stopping state in our problems.

Note that every point in the regions D through R_{m-1} represents in fact an intermediate state of a valid simulation of a g sequence, while all other points map immediately to the origin. Thus, f globally converges to zero if and only if it is mortal if and only if g satisfies the GCP. ◀

In the following sections we prove results which are strictly stronger than Theorem 7, and their proofs are also more involved, and could not be fully given in this extended abstract; hopefully, the above proof was sufficiently clear, and the proof fragments in the sequel will give the reader an idea of the techniques that were necessary to obtain the other results.

3 Undecidability with a Bounded Number of Regions

The number of regions in the definition of function f above depends on the modulus m of the Collatz function. In this section, we establish that the mortality problem is also hard when the number of regions is bounded by some *a priori* constant (if it is large enough). This will follow immediately from proving that a result like Theorem 6 holds for a fixed modulus (which may be interesting in itself)[1].

First, we introduce a special variant of the counter machine. An *enhanced CM* is a counter machine, where an instruction may increase the value of a register by an arbitrary positive constant (a decrease, however, is limited to 1 as usual).

▶ **Theorem 8.** *From an (ordinary) counter machine M, one can compute an enhanced counter machine U_M such that U_M is mortal if and only if M halts on every initial state $(1, \langle x, 0, \ldots, 0\rangle)$. The number of registers and instructions of U_M is independent of M.*

The proof combines the reduction of totality to mortality by [3] with the use of a universal CM. The enhanced instruction set is necessary to keep the size of U_M independent of M.

▶ **Theorem 9.** *There is a constant m such that GCP restricted to modulus m is Π^0_2-complete.*

Proof. We reduce from the standard problem: Given an (ordinary) counter machine M, does it halt on every initial state $(1, \langle x, 0, \ldots, 0\rangle)$? The reduction first maps M to U_M, then translates this counter machine to a generalized Collatz function g. The modulus of g only depends on the number of registers and instructions in U_M, and is independent of M. ◀

▶ **Corollary 10.** *There is a constant m such that global convergence and mortality over $\mathbb{Z}$ of piecewise affine functions f with integer coefficients and m regions are Π^0_2-complete.*

It is natural to ask what the threshold of undecidability is regarding the number of regions. Consider the function f defined in Section 2. How many regions does it have? There are $m + 2$ rows in the table. But the last one does not count as a single region according to Definition 3. The problem is that it is not convex, and therefore has to be split into convex polyhedra. It seems natural to adjust the way we count regions by allowing the function to be defined explicitly on certain convex polyhedral regions, and *zero elsewhere*. We are interested in the number $\mathcal{R}$ of these convex regions. The case $\mathcal{R} = 1$ coincides with an important open problem, see [5]. For $\mathcal{R} = 2$ the problem has been recently shown to be undecidable [2].

---

[1] By constructing a universal GCP, both [12, 7] show that undecidability of the *reachability problem* (is 1 reachable from a given initial point?) holds for a fixed modulus. But this does not imply any conclusion for mortality; universal machines are, of course, immortal.

4 Undecidability for Monic Functions

The result of the last section may be interpreted as showing that the hardness of the problem does not depend on allowing an unbounded number of regions. So, it must come from allowing an unbounded set of possible affine functions for each region. In this section we will show that even when restricting the coefficients in the linear components of these functions to 1, we still have undecidability.

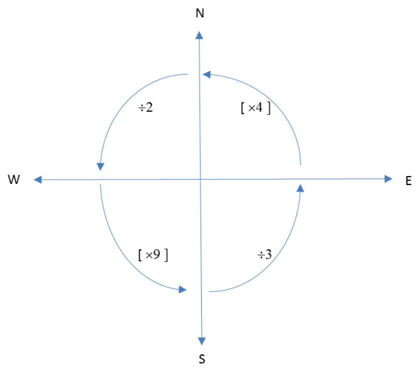

Figure 1 The dynamics of a Compass Collatz-like function.

▶ **Definition 11.** A *monic* piecewise affine function (abbreviated to "monic PAF") on $\mathbb{Z}^2$ is a piecewise affine function where each of the defining affine components has the form $f(x) = (x_{i_1} + b_1, x_{i_2} + b_2)$, where $\{i_1, i_2\} = \{1, 2\}$. In addition, the halfspaces which define the space partition are given by inequalities of the form $cx_i + d \geq 0$ such that $c = \pm 1$.

The main point is that the definition of the function f does not allow for terms ax_j with $a > 1$ (or sums like $x_1 + x_2$).

In the next proof we will use 2-counter machines (2CM). By combining the technique of Blondel et al. with that of Simon and Kurtz, it is not hard to show that 2CM mortality is Π^0_2-complete. It will be useful to restrict the machines under consideration to a certain class of "normal forms". A normalized machine: (1) modifies (increments or decrements) every register in every transition; (2) only halts at a state where the values of the counters are either $\langle 0, 0\rangle$, $\langle 0, 1\rangle$ or $\langle 1, 0\rangle$.

From these machines, we construct a special (and new) variant of the Collatz problem, the *Compass Collatz-like function*. In describing such functions we make use of the set $\mathbf{C} = \{E, N, W, S\}$ of Compass Directions. A pair (x, Δ), where $x \in \mathbb{N}$, may be depicted as a point on one of the axes in the Cartesian plane, in the direction Δ (we may also refer to such a point as lying *on the* Δ *axis*). We refer to it as *a compass point.*

▶ **Definition 12.** A function $g : \mathbb{N}_+ \times \mathbf{C} \to \mathbb{N}_+ \times \mathbf{C}$ is called a *Compass Collatz-like function* if there is a number $m = 6p$ with $p \geq 5$ a prime, sets $R_N, R_S \subseteq [0, m-1]$ and integers $w_i \in [0, m-1]$ for $i = 0, \ldots, m-1$, such that g satisfies the following equations (for convenience we represent its argument in the form $mx + rp + i$, where $x \geq 0$, $0 \leq r < 6$ and $0 \leq i < p$):

$$
\begin{aligned}
g(mx+rp+i, E) &= \begin{cases} (mx+rp+i, N) & rp+i \in R_N \\ (4(mx+rp)+i, N) & rp+i \notin R_N \end{cases} \\
g(mx+rp+i, N) &= (\frac{1}{2}mx + \lfloor \frac{1}{2} r \rfloor p + i, W) \\
g(mx+rp+i, W) &= \begin{cases} (mx+rp+w_{rp+i}, S) & rp+i \in R_S \\ (9(mx+rp)+w_{rp+i}, S) & rp+i \notin R_S \end{cases} \\
g(mx+rp+i, S) &= (\frac{1}{3}mx + \lfloor \frac{1}{3} r \rfloor p + i, E)
\end{aligned}
\tag{2}
$$

The action of such a function is schematically represented in Figure 1. The "compass" representation is useful for the transition to a 2-dimensional dynamical system (and also makes it easier to visualize the dynamics).

▶ **Definition 13.** CCP (for Compass Collatz-like Problem) is the problem of deciding, from a standard representation of g, whether it is the case that every such sequence reaches some point (x, E) with $x < m$ (the set of such points is called *the final zone*).

▶ **Lemma 14.** *CCP is Π^0_2-complete.*

Proof. (Sketch) The proof is a reduction from 2CM mortality. Given a normal 2-counter machine M we simulate it by a Compass Collatz-like function, based on the following outline.

Let $m = 6p$ where $p > 3$ is a prime such that the number of internal states of M is $p - 1$ (there is no loss of generality). We represent a state $s = (i, \langle r_1, r_2\rangle)$ by $\widehat{s} = (2^{r_1}3^{r_2}p + i)$. We design the function g so that it takes every encoded state $(\widehat{s}, E)$, in a bounded number of steps, to a $(\widehat{s'}, E)$ where s' is the successor state to s.

In greater detail: first, the point will move to the North axis, possibly increasing r_1 by two; then, move to the West axis, while decreasing r_1 by one; the result is that r_1 has either been incremented or decremented—according to the instruction i, of course. In a similar way, moving to the South axis and subsequently to the East again, an increment or decrement of the second register is simulated.

If M is mortal, every trajectory starting at any encoded state will arrive at a *halting configuration*, represented by a number of the form $(rp + 0, E)$ with $r \in \{1, 2, 3\}$, so the CCP is satisfied. If M is not mortal, it has a non-ending computation, which translates into an infinite trajectory under g.

Since g has to be total, we have to define g for points which do not encode a state, or appear on the trajectory from a state to its successor, as described above; the full proof (omitted here, for lack of space) includes a complete definition of g and argues that our claim that the CCP is satisfied if M is mortal holds when taking all points into account. ◀

▶ **Theorem 15.** *Global convergence and mortality of monic piecewise-affine functions over $\mathbb{Z}^2$ are Π^0_2-complete problems.*

The theorem is proved by reducing the CCP to the mortality problem for monic PAFs. The details of this construction could not fit in this abstract; but the reader may be able to gain an idea of how it works from the following example, where the classic $3x+1$ problem is represented by a monic 2-dimensional PAF, whose iteration reaches $(1, 0)$ from initial point $(x, 0)$ if and only if the Collatz sequence from x reaches 1.

Recall that in the $3x + 1$ problem there are just two possible "updates," division by 2 and the mapping $x \mapsto 3x + 1$. This makes it simpler than the Compass problem, where there are four "updates" and a large modulus. However, we still make use of trajectories that orbit around the origin, starting with the Cartesian point $(x, 0)$ (Figure 2). The NE quadrant carries $(x, 0)$ to $(\lfloor x/2\rfloor, x \bmod 2)$; if the division was even, the point is mapped to

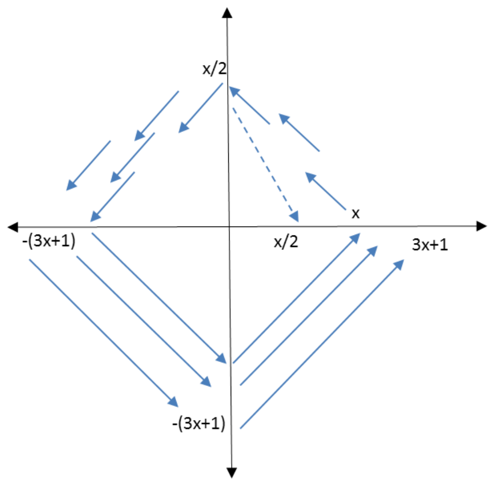

Figure 2 Simulating the $3x+1$ function by a 2-dimensional monic PAF.

$(x/2, 0)$, ready for the next iteration; otherwise, proceeding counter-clockwise, this point is carried first to $(-(3x+1), 0)$, then to $(0, -(3x+1))$ and finally to $(3x+1, 0)$.

Here is the complete definition of the PAF:

role of region	constraints	$f(x,y)$	region function	constraints	$f(x,y)$
Div by 2	$x{\geq}2,\ y \geq 0$	$(x-2, y+1)$	Compute $3x+1$	$x{<}0\ ,\ y \geq 1$	$(x-6, y-1)$
$x \bmod 2 = 0$	$x{=}0,\ y \geq 0$	(y, x)	West to South	$x{<}0\ ,\ y < 0$	$(x+1, y-1)$
$x \bmod 2 = 1$	$x{=}1,\ y \geq 1$	$(x-5, y)$	South to East	$x{\geq}0\ ,\ y < 0$	$(x+1, y+1)$
			Final zone	$0 \leq x \leq 1,\ y = 0$	(x, y)

5 Decidability and Complexity in One Dimension

Blondel et al. prove that global convergence is decidable for $n = 1$ when the function is continuous. We prove the result for the integers, where continuity is irrelevant. Furthermore, we identify the complexity of the problem, proving it to be PSPACE-complete.

The following examples hint at the difference between mortality over the integers and over the reals. The first function has a fixed point at 10/3 and therefore is not mortal over the reals (or rationals); but it is over the integers.

$$f_1(x) = \begin{cases} -2x+10 & x > 2 \\ 4 & x < 0 \\ 0 & x = 0 \\ 4-2x & 1 \leq x \leq 2 \end{cases} \qquad f_2(x) = \begin{cases} x-3 & x > 2 \\ 4 & x < 0 \\ 0 & x = 0 \\ 4-2x & 1 \leq x \leq 2 \end{cases}$$

Below, we give an algorithm to decide mortality over the integers. We also show that it requires polynomial space (Remark: space complexity is interpreted in the standard way, that is, number of bits used as a function of the number of bits in the input).

Note that in dimension one, the regions are just a partition of $\mathbb{Z}$ into a finite number of intervals. We may assume that these are given explicitly as the list of the end points of closed intervals, e.g.,

$$(-\infty, -3], [-2, 0], [1, +\infty)\,.$$

There will always be one interval infinite to the left, which we denote by $(-\infty, L]$, and one infinite to the right, denoted by $[R, +\infty)$; and by breaking intervals into parts if necessary we can ensure $L < 0$ and $R > 0$. We denote the function in the negative part by $a^-x + b^-$ and the function in the positive part by $a^+x + b^+$.

We may also assume $f(0) = 0$ since, for the convergence problem, the answer is negative if this does not hold, while for mortality, we may as well modify the function so that $f(0) = 0$.

First, we define

$$\rho_0 = \max(\{R\} \cup \{f(x), \text{ where } L \leq x \leq R\})$$
$$\lambda_0 = \min(\{L\} \cup \{f(x), \text{ where } L \leq x \leq R\})$$

We note that this value can be easily calculated in polynomial time, since for each finite interval, $f(x)$ assumes the maximum (or minimum) at one of its ends. These values show how far away from 0 one can get without using the infinite regions.

Next, we show that one can efficiently either determine that the dynamic system is *divergent*, which means that there is an unbounded trajectory; or find a finite attractor, an interval such that all trajectories eventually stay within it.

▶ **Lemma 16.** *Suppose that at least one of a^+, a^- is non-negative. If either a^+ or a^- is bigger than 1; or $a^+ = 1$ and $b^+ \geq 0$; or $a^- = 1$ and $b^- \leq 0$, then f is divergent, and not mortal. Otherwise, it has the finite attractor $A = [\lambda, \rho]$ where*

$$\rho = \max(\rho_0,\ (\lambda_0 - |b^-|) \cdot \min(a^-, 0))$$
$$\lambda = \min(\lambda_0,\ (\rho_0 + |b^+|) \cdot \min(a^+, 0))\,.$$

The algorithm for analysing mortality is now as follows. First, if both a^+ an a^- are negative, we construct (in polynomial time) a representation of $f \circ f$, whose asymptotic behaviour is the same, and has positive coefficients in the infinite regions; we thus assume that Lemma 16 is applicable. By testing the conditions stated in the lemma, we either conclude immediately that f is not mortal, or get the attactor A. In the latter case, we now proceed to trace, for each point in this interval, the trajectory from this point, until finding that it reaches zero, or that it cycles without meeting the origin—so we know if f is mortal. It is not hard to verify that this can be accomplished in polynomial space. We conclude

▶ **Theorem 17.** *Global convergence over $\mathbb{Z}$ of a piecewise affine function f with integer coefficients is a PSPACE problem.*

It may be interesting to note that the decision procedure for the one-dimensional case in [3] is much simpler, and in fact polynomial-time (for rational coefficients in a standard representation). Keep in mind, however, that it solves a different problem (a continuous state space and a continuous function), which is neither a sub-problem nor a super-problem of the problem we consider. In fact, for our problem, our problem turns out to be PSPACE-hard. To prove it, we next introduce several Turing-machine variants as intermediate representations, and show a sequence of reductions, starting with a standard PSPACE-hard problem and culminating in our mortality problem (in this extended abstract, the more technical steps have been omitted to save space).

The first machine is a restricted form of *linearly bounded automaton* (LBA) [21].

▶ **Definition 18.** A *tally LBA* is a single-tape machine that receives as input a string of the form 0^n for some $n \geq 0$; this string is initially written on its work tape, delimited by endmarkers. The machine is guaranteed to never move beyond the endmarkers. The tape alphabet is binary.

In the next definition, we consider the tally string to be part of the machine's description: hence, a given machine only performs a single computation.

▶ **Definition 19.** A *fixed-space machine with oblivious queue access* is a Turing machine with the following features:

1. The work tape is a *queue* of a fixed capacity n (which we consider to be given, in tally form, as part of the standard description of such a machine). In every step the machine strips a symbol from the front of the queue and adds one to the rear. Hence, an instruction of the machine is given by a 4-tuple (q, b, q', b'), where:
   q is the current control state, q' the next;
   b the symbol read off the queue, b' the symbol appended to the queue.
   We use the customary symbol δ for the set of these 4-tuples.
2. The work-tape (queue) alphabet is binary.
3. The initial contents of the queue are 0^n.

▶ **Definition 20.** A *fixed-space machine with oblivious queue access and a clock*, briefly a fixed-space Q&C machine, is a Turing machine with the following features:

1. The work tape is a *queue*, as in the previous definition.
2. The machine also has a *clock*. This device is a counter (a register of non-negative integer value) that is automatically decremented each time the machine has completed another cycle through the queue (that is, exactly n transitions). If the clock reaches zero, the machine is reset: the queue is cleared, the control state is reset to an initial state (0) and the clock is reset to its initial value.
3. The initial contents of the queue are 0^n; the initial clock value is given—in binary—as part of the standard description of such a machine.

▶ **Lemma 21.** *The halting problem of tally LBAs is a PSPACE-hard problem.*

▶ **Lemma 22.** *The computation of a tally LBA on input 0^n can be simulated by a fixed-space machine with oblivious queue access, starting on a blank queue of capacity $2n$.*

Mortality of fixed-space machines is defined as usual: halting when started with any possible configuration. Note that there are finitely many such configurations, due to the fixed space.

▶ **Lemma 23.** *The halting problem for fixed-space Turing machines with oblivious queue access can be reduced, in logarithmic space, to mortality of a fixed-space Q&C machine.*

Proof. If the given machine ever halts, the length of its computation must be bounded by $2^n \cdot m$, where n is the queue size and m the number of control states. We let this be the initial value of the clock, so that if the machine does halt, it will halt before the clock times out. If the machine does not halt, it will eventually time out, then restart, ad infinitum. To see that a halting machine is transformed into a mortal one, note that regardless of its initial configuration, the machine will either halt or time out, and from that point on, faithfully simulate the given input-free machine. ◀

▶ **Lemma 24.** *Mortality of fixed-space Turing machines with oblivious queue access is PSPACE-hard.*

▶ **Theorem 25.** *Global convergence over $\mathbb{Z}$ of a piecewise affine function f with integer coefficients is PSPACE-hard (under logspace reductions).*

Proof. We reduce from mortality of fixed-space machines with oblivious queue access. The reduction transforms a machine M, given with its space bound n, into a representation of a piecewise-affine function f that simulates M and is mortal if and only if M is.

Let us first describe the essence of this simulation. Suppose that M has m states. A configuration of M is specified by (q, w) where $q \in [0, m-1]$ is the control state, and $w \in \{0,1\}^n$ is the contents of the queue.

By identifying w with the integer that it represents in binary notation, we define an integer that encodes a configuration:

$$\langle q, w \rangle = q \cdot 2^n + w\,. \tag{3}$$

Next, we define f as a function that maps a configuration to the next, simulating the machine; we present the definition by cases.

For each $(q, b, q', b') \in \delta$, we define $\quad f(\langle q, bz \rangle) = \langle q', zb' \rangle\,. \quad (4)$

If there is no transition starting with (q, b), $\quad f(\langle q, bz \rangle) = 0. \quad (5)$

We now verify that the above definitions yield a piecewise-affine function. Since this is a one-dimensional PAF, its regions of definition are intervals and we use the notation $a + [b, c]$ for $[a + b, a + c]$. Let $R_{q,b}$ be the interval $q \cdot 2^n + b \cdot 2^{n-1} + [0, 2^{n-1} - 1]$. The function f is defined on each such interval according to the applicable case; specifically, every transition described by (4) is translated into: $x \in R_{q,b} \Rightarrow f(x) = 2(x - q \cdot 2^n - b \cdot 2^{n-1}) + q' \cdot 2^n + b'$ while if there is no such transition, $x \in R_{q,b} \Rightarrow f(x) = 0$. ◀

6 Similar Problems

There are several natural problems similar to mortality and global convergence, and results fall out of the same reductions (mostly). Here are two examples. The first behaves just the same as mortality (Π^0_2-complete in two dimensions, PSPACE-complete in one), while the second differs in the one-dimensional case being in PTIME.

Global convergence to a fixed point [13].
Input: f, a PAF. Question: do all trajectories of f reach a fixed point?

Convergence to a Finite Set (In Dynamical System parlance: *Finite Attractor*).
Input: f, as previously. Question: is there a finite set S such that every sequence $x_{t+1} = f(x_t)$ stays within S for all t large enough?

7 Conclusion and Open Problems

We have presented work on the border between Dynamical System Theory (much of which refers to continuous state-spaces) and the Theory of Computation in discrete models. In particular, this work connects Dynamical System Theory to problems of program termination. I'd like to argue that such results may be of interest in the context of continuous-space dynamical systems (e.g., as models of some kinds of physical systems), since, unlike previous proofs, the undecidability is not derived from the encoding of information into the fractional bits of a value, which boils down to assuming unlimited precision in measurement.

The focus of this work has been on deciding global properties of systems, so we have not dwelled on the problem that corresponds to the common Turing-machine *halting problem*, namely: given a function and an initial value x_0, does the sequence beginning with x_0 reach zero. However it is easy to see, from our proofs, that this problem is RE-complete in two dimensions, and PSPACE-complete in one. The problem: does the sequence beginning with x_0 reach a given value y? Is known as the *orbit problem* and was solved in polynomial time for linear transformations over $\mathbb{Q}^n$ [11].

I hope that the results presented are of interest, but also the techniques and connections made to Collatz-like problems and to automata that capture PSPACE. Finally, here are some open problems.

1. Is there any constant bound on the number of regions which suffices to make mortality of monic piecewise-affine functions over $\mathbb{Z}^2$ as hard as the general problem, i.e., Π^0_2 hard?
2. (Braverman's problem) Is mortality decidable for functions which are zero outside a single convex region (and affine within)?
3. What is the complexity of the mortality problem in the one-dimensional case, when the coefficient of x is always 1?
4. What is the complexity of the mortality problem in the one-dimensional case, when the number of intervals is considered a constant?

Acknowledgement. I thank Jakob Grue Simonsen for a helpful discussion.

References

1. Amir M. Ben-Amram and Samir Genaim. On the linear ranking problem for integer linear-constraint loops. In *Proceedings of the 28th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. ACM press, 2013.
2. Amir M. Ben-Amram, Samir Genaim, and Abu Naser Masud. On the termination of integer loops. *ACM Transactions on Programming Languages and Systems*, to appear.
3. Vincent D. Blondel, Olivier Bournez, Pascal Koiran, Christos H. Papadimitriou, and John N. Tsitsiklis. Deciding stability and mortality of piecewise affine dynamical systems. *Theor. Comput. Sci.*, 255(1-2):687–696, 2001.
4. Vincent D. Blondel and John N. Tsitsiklis. A survey of computational complexity results in systems and control. *Automatica*, 36(9):1249–1274, 2000.
5. Mark Braverman. Termination of integer linear programs. In Thomas Ball and Robert B. Jones, editors, *Computer Aided Verification, 18th International Conference, CAV 2006*, volume 4144 of *Lecture Notes in Computer Science*, pages 372–385. Springer, 2006.
6. Michael Brin and Garrett Stuck. *Introduction to dynamical systems.* Cambridge University Press, Cambridge, UK, 2002.
7. Serge Burckel. Functional equations associated with congruential functions. *Theoretical Computer Science*, 123(2):397–406, January 1994.
8. Michael Colón and Henny Sipma. Synthesis of linear ranking functions. In *7th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 2031 of *Lecture Notes in Computer Science*, pages 67–81. Springer, 2001.
9. John. H. Conway. Unpredictable iterations. In *Proc. 1972 Number Theory Conf., Univ. Colorado, Boulder*, pages 49–52. 1972. Reprinted with historical commentary in [15].
10. John. H. Conway. FRACTRAN: a simple universal programming language for arithmetic. In T. M. Cover and B. Gopinath, editor, *Open Problems in Communication and Computation*, pages 3–27. Springer-Verlag, 1987. Reprinted with historical commentary in [15].
11. R. Kannan and R. J. Lipton. Polynomial-time algorithm for the orbit problem. *J. ACM*, 33(4):808–821, October 1986.
12. František Kaščák. Small universal one–state linear operator algorithm. In Ivan M. Havel and Vaclav Koubek, editors, *Mathematical Foundations of Computer Science (MFCS '92)*, volume 629 of *LNCS*, pages 327–335, Springer, 1992.
13. Pascal Koiran, Michel Cosnard, and Max Garzon. Computability with low-dimensional dynamical systems. *Theor. Comput. Sci.*, 132:113–128, September 1994.
14. Stuart A. Kurtz and Janos Simon. The undecidability of the generalized Collatz problem. In Jin-Yi Cai, S. Barry Cooper, and Hong Zhu, editors, *Theory and Applications of Models of Computation, 4th International Conference, TAMC 2007, Shanghai, China, May 22-25, 2007*, volume 4484 of *Lecture Notes in Computer Science*, pages 542–553. Springer, 2007.
15. Jeffrey C. Lagarias. *The Ultimate Challenge: The* $3x+1$ *Problem.* American Mathematical Society, 1st edition, 2010.
16. Marvin L. Minsky. *Computation: finite and infinite machines.* Prentice-Hall, USA, 1967.
17. Cristopher Moore. Unpredictability and undecidability in dynamical systems. *Phys. Rev. Lett.*, 64(20):2354–2357, May 1990.
18. Andreas Podelski and Andrey Rybalchenko. A complete method for the synthesis of linear ranking functions. In *VMCAI*, pages 239–251, 2004.
19. H.T. Siegelmann and E.D. Sontag. Analog computation, neural networks, and circuits,. *Theor. Comp. Sci.*, 131:331–360, 1994.
20. Ashish Tiwari. Termination of linear programs. In Rajeev Alur and Doron Peled, editors, *Computer Aided Verification*, volume 3114 of *Lecture Notes in Computer Science*, pages 387–390. Springer Berlin / Heidelberg, 2004.
21. K. Wagner and G. Wechsung. *Computational Complexity.* D. Reidel, Dordrecht, 1986.