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

Amir M. Ben-Amram

The Academic College of Tel-Aviv Yaffo

STACS 2013
Background: Dynamical systems

Discrete-time dynamical system

A state space $X$ and an evolution function $f : X \rightarrow X$.

Typical spaces: $\mathbb{R}^n$ (continuous), $\mathbb{Z}^n$ (discrete)
Background: Dynamical systems

Discrete-time dynamical system

- a state space $X$ and an evolution function $f : X \rightarrow X$.

Typical spaces: $\mathbb{R}^n$ (continuous), $\mathbb{Z}^n$ (discrete)

Trajectory (orbit)

- the sequence $x_{t+1} = f(x_t)$, for some initial point $x_0 \in X$. 

![Diagram of discrete-time dynamical system]

- Amir Ben-Amram

Iterated Piecewise Affine Functions over the Integers
Background: Dynamical systems

Discrete-time dynamical system

A **state space** $X$ and an **evolution function** $f : X \rightarrow X$.

Typical spaces: $\mathbb{R}^n$ (continuous), $\mathbb{Z}^n$ (discrete)

Trajectory (orbit)

The sequence $x_{t+1} = f(x_t)$, for some initial point $x_0 \in X$.

In Dynamical System theory, we are often interested in **asymptotic** properties of *all trajectories*. 
A dynamical system $f$ is **mortal** if every trajectory reaches a designated final point (w.l.o.g., 0).

— this program always halts:

```
while $x \neq 0$ do
    $x \leftarrow f(x)$
```
Mortality

Definition

A dynamical system $f$ is mortal if every trajectory reaches a designated final point (w.l.o.g., 0).

— this program always halts:

```plaintext
while $x \neq 0$ do
  $x \leftarrow f(x)$
```

Program termination is a basic and important problem of software verification. Much of the effort goes into analysing “simple while loops”
A dynamical system $f$ is **mortal** if every trajectory reaches a designated final point (w.l.o.g., 0).

— this program always halts:

```plaintext
while $x \neq 0$ do
  $x \leftarrow f(x)$
```

Program termination is a basic and important problem of software verification.

Much of the effort goes into analysing “simple while loops”
A dynamical system $f$ is **mortal** if every trajectory reaches a designated final point (w.l.o.g., 0).

— this program always halts:

```plaintext
while $x \neq 0$ do
  $x \leftarrow f(x)$
```

Program termination is a basic and important problem of software verification.

Much of the effort goes into analysing “simple while loops”

Halting problems are undecidable in general
The **halting problem** considers halting of a given program with given input.

- The **halting problem** is undecidable — but RE \( \Sigma^0_1 \)

**Termination** (in program verification) is called **Totality** in Recursion Theory.

- the program should halt when initialized with any input
- harder than the halting problem: \( \Pi^0_2 \)-complete

**Mortality** is more stringent: A computation started at *any state* should halt.

- Kurtz and Simon (2007): Counter Machine mortality is \( \Pi^0_2 \)-complete.
Some simple classes of functions have been studied both in Dynamical System Theory and in Program Verification.

- Affine functions: \( f(\vec{x}) = A\vec{x} + b \)

- Piecewise affine functions. The space is divided (by hyperplanes) into a finite number of regions, and \( f \) is affine on each region.

\[
\begin{align*}
  f(\vec{x}) &= A_1\vec{x} + b_1 \\
  f(\vec{x}) &= A_2\vec{x} + b_2 \\
  f(\vec{x}) &= A_3\vec{x} + b_3 \\
  f(\vec{x}) &= A_4\vec{x} + b_4
\end{align*}
\]
Is Mortality decidable for simple functions?

(Note: answer may depend on domain—$\mathbb{R}^n$, $\mathbb{Z}^n$.)

For affine functions, mortality is decidable, using linear-algebraic methods (both domains).

Piecewise-affine functions were studied in Blondel, Bournez, Koiran, Papadimitriou & Tsitsiklis (2001).

Mortality is undecidable for piecewise-affine functions (with rational coefficients) over $\mathbb{R}^n$ for all $n \geq 2$. 

[BBKPT 2001], Theorem 2
Is Mortality decidable for simple functions?

(Note: answer may depend on domain—$\mathbb{R}^n$, $\mathbb{Z}^n$.)

For affine functions, mortality is **decidable**, using linear-algebraic methods (both domains).
Is Mortality decidable for simple functions?

(Note: answer may depend on domain—$\mathbb{R}^n$, $\mathbb{Z}^n$.)

- For affine functions, mortality is **decidable**, using linear-algebraic methods (both domains).
- Piecewise-affine functions were studied in Blondel, Bournez, Koiran, Papadimitriou & Tsitsiklis (2001).
Is Mortality decidable for simple functions?

(Note: answer may depend on domain—\( \mathbb{R}^n, \mathbb{Z}^n \).)

- For affine functions, mortality is decidable, using linear-algebraic methods (both domains).
- Piecewise-affine functions were studied in Blondel, Bournez, Koiran, Papadimitriou & Tsitsiklis (2001).

[BBKPT 2001], Theorem 2

Mortality is undecidable for piecewise-affine functions (with rational coefficients) over \( \mathbb{R}^n \) for all \( n \geq 2 \).

[BBKPT 2001], Corollary 1

Mortality is decidable for continuous piecewise-affine functions in one dimension.
Is Mortality decidable for simple functions?

(Note: answer may depend on domain—$\mathbb{R}^n$, $\mathbb{Z}^n$.)

- For affine functions, mortality is **decidable**, using linear-algebraic methods (both domains).
- Piecewise-affine functions were studied in Blondel, Bournez, Koiran, Papadimitriou & Tsitsiklis (2001).

[[BBKPT 2001], Theorem 2]

Mortality is **undecidable** for piecewise-affine functions (with rational coefficients) over $\mathbb{R}^n$ for all $n \geq 2$.

Reduction from **halting**
Is Mortality decidable for simple functions?

(Note: answer may depend on domain—$\mathbb{R}^n$, $\mathbb{Z}^n$.)

- For affine functions, mortality is **decidable**, using linear-algebraic methods (both domains).
- Piecewise-affine functions were studied in Blondel, Bournez, Koiran, Papadimitriou & Tsitsiklis (2001).

[BBKPT 2001], Theorem 2

Mortality is **undecidable** for piecewise-affine functions (with rational coefficients) over $\mathbb{R}^n$ for all $n \geq 2$.

Reduction from **halting**

“Reductions usually encode machine states as points in the space. Not all points represent states, which makes a reduction to mortality difficult.”
Meanwhile, in Program Termination…

Very similar questions have been asked (and partly answered)

- (Un)Decidability results:
  - Tiwari — CAV 2004
  - Braverman — CAV 2006
  - Ben-Amram, Genaim, Masud — VMCAI 2011

- Heuristic solutions (many)

The state space in this area is typically discrete ($\mathbb{Z}^n$)
Meanwhile, in Program Termination…

Very similar questions have been asked (and partly answered)

- (Un)Decidability results:
  - Tiwari — CAV 2004
  - Braverman — CAV 2006
  - Ben-Amram, Genaim, Masud — VMCAI 2011

- Heuristic solutions (many)

The state space in this area is typically discrete \((\mathbb{Z}^n)\)

Goal of this work

Characterize decidability of mortality for functions over \(\mathbb{Z}^n\)
(in particular, in terms of dimension).
Consider piecewise-affine functions over $\mathbb{Z}^n$.

- Mortality is undecidable for $n = 2$.

- Mortality is decidable for $n = 1$. 
Consider piecewise-affine functions over $\mathbb{Z}^n$.

- Mortality is $\Pi^0_2$ complete for $n = 2$.

- Mortality is PSPACE-complete for $n = 1$. 
Consider piecewise-affine functions over $\mathbb{Z}^n$.

- Mortality is $\Pi^0_2$ complete for $n = 2$.
- Also when the number of regions is bounded by some constant.

- Mortality is PSPACE-complete for $n = 1$. 

Results
Consider piecewise-affine functions over $\mathbb{Z}^n$.

- Mortality is $\Pi_2^0$ complete for $n = 2$.
  - Also when the number of regions is bounded by some constant.
  
  Also when the affine functions are monic:
  
  $f(x, y) = (x + a, y + b)$  or  $f(x, y) = (y + b, x + a)$

  No multiplication involved

- Mortality is PSPACE-complete for $n = 1$. 

Methods (undecidability proofs)
Methods (undecidability proofs)

Dynamical Systems

Mortality

Program Termination
Methods (undecidability proofs)

- Dynamical Systems
- Program Termination
- Mortality
- Collatz problems
The Collatz Problem

Consider the function over $\mathbb{N}$:

$$g(x) = \begin{cases} 
3x + 1 & \text{if } x \mod 2 = 1 \\
\frac{x}{2} & \text{if } x \mod 2 = 0 
\end{cases}$$

Problem (posed by Lothar Collatz)

Do all trajectories of this function (over $\mathbb{N}$) converge to 1?

Example: $5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$

- Celebrated open problem
The Collatz Problem

Consider the function over $\mathbb{N}$:

$$g(x) = \begin{cases} 
3x + 1 & \text{if } x \mod 2 = 1 \\
\frac{x}{2} & \text{if } x \mod 2 = 0 
\end{cases}$$

Problem (posed by Lothar Collatz)
Do all trajectories of this function (over $\mathbb{N}$) converge to 1?

Example: $5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$
- Celebrated open problem

**Definition**

A **generalized Collatz function** has the form

\[
g(x) = \begin{cases} 
  a_0y + b_0 & \text{if } x = my + 0 \\
  a_1y + b_1 & \text{if } x = my + 1 \\
  \vdots \\
  a_{m-1}y + b_{m-1} & \text{if } x = my + (m-1)
\end{cases}
\]

For some modulus \( m > 0 \) and \( a_0, b_0, \ldots, a_{m-1}, b_{m-1} \geq 0 \).

**GCP** (Generalized Collatz Problem) is the problem of deciding whether every trajectory of \( g \) reaches 1.
Theorem (Kurtz & Simon 2007, extending Conway)

\textit{GCP is }$\Pi^0_2$\textit{-complete.}

Proof: reduction from the GCP.

Computing $g(x)$ involves: (1) division by $m$, (2) application of the function $f_i(y) = a_iy + b_i$ according to the remainder $i$.

The division is simulated by a trajectory in two dimensions.
Theorem (Kurtz & Simon 2007, extending Conway)

\[ GCP \text{ is } \Pi^0_2\text{-complete.} \]

Theorem (new)

\emph{Mortality of piecewise affine functions over } \mathbb{Z}^2 \emph{ is } \Pi^0_2 \emph{ complete.}
Theorem (Kurtz & Simon 2007, extending Conway)

\(GCP \text{ is } \Pi^0_2\)-complete.

Theorem (new)

Mortality of piecewise affine functions over \(\mathbb{Z}^2\) is \(\Pi^0_2\) complete.

**Proof:** reduction from the GCP.

Computing \(g(x)\) involves: (1) division by \(m\), (2) application of the function \(f_i(y) = a_iy + b_i\) according to the remainder \(i\).

The division is simulated by a trajectory in two dimensions.
Simulating a Collatz problem in $\mathbb{Z}^2$

$$f(0, y) = (y/2, 0)$$

$$f(1, y) = (6y + 4, 0)$$

$$g(x) = \begin{cases} 
3x + 1 & \text{if } x \mod 2 = 1 \\
 x/2 & \text{if } x \mod 2 = 0 
\end{cases}$$

$$f(x, y) = (x - 2, y + 1)$$
Recall that a monic function has the form
\[ f(x, y) = (x + a, y + b) \text{ or } f(x, y) = (y + b, x + a). \]

This prevents us from directly applying the function
\[ f_i(y) = a_i y + b_i. \]

We have to simulate multiplication as well.
Simulating the $3x + 1$ function

\[ f(x, y) = (x - 2, y + 1) \]
Simulating the $3x + 1$ function

\[ f(x, y) = (x - 6, y - 1) \]

\[ f(x, y) = (x - 2, y + 1) \]
Simulating the $3x + 1$ function

\[ f(x, y) = (x - 6, y - 1) \]

\[ f(x, y) = (x - 2, y + 1) \]

\[ f(x, y) = (y, x) \]
Simulating the $3x + 1$ function

\[ f(x, y) = (x - 6, y - 1) \]
\[ f(x, y) = (x - 2, y + 1) \]
\[ f(x, y) = (y, x) \]
\[ f(x, y) = (x + 1, y + 1) \]
Simulating general Collatz functions?

- I see no (direct) way to do it
Simulating general Collatz functions?

- I see no (direct) way to do it
- Instead, we introduce the Compass Collatz Functions
Compass Collatz Functions

Let $\mathcal{C} = \{E, N, W, S\}$.
The set of compass points is $\mathcal{P} = \mathbb{N} \times \mathcal{C}$.
A Compass Collatz function is $g : \mathcal{P} \to \mathcal{P}$ such that:
Compass Collatz Functions

Let $\mathcal{C} = \{E, N, W, S\}$.

The set of compass points is $\mathcal{P} = \mathbb{N} \times \mathcal{C}$.

A Compass Collatz function is $g : \mathcal{P} \to \mathcal{P}$ such that:

there is a number $m = 6p$ with $p \geq 5$ a prime, sets $R_N, R_S \subseteq [0, m-1]$ and integers $w_i \in [0, m-1]$, so that if $x = mx + rp + i$, where $0 \leq r < 6, 0 \leq i < p$:

\[
\begin{align*}
    g(mx + rp + i, E) &= \begin{cases} 
        (mx + rp + i, N) & rp + i \in R_N \\
        (4(mx + rp) + i, N) & rp + i \notin R_N
    \end{cases} \\
    g(mx + rp + i, N) &= \left(\frac{1}{2}mx + \left\lfloor \frac{1}{2}r \right\rfloor + i, W\right) \\
    g(mx + rp + i, W) &= \begin{cases} 
        (mx + rp + w_{rp+i}, S) & rp + i \in R_S \\
        (9(mx + rp) + w_{rp+i}, S) & rp + i \notin R_S
    \end{cases} \\
    g(mx + rp + i, S) &= \left(\frac{1}{3}mx + \left\lfloor \frac{1}{3}r \right\rfloor + i, E\right)
\end{align*}
\]
Compass Collatz Functions

\[ f(x, y) = (x - 1, y - 2) \]

\[ f(x, y) = (x - 1, y + 4) \]

\[ f(x, y) = (x + 1, y - 9) \]

\[ f(x, y) = (x + 1, y + 3) \]
Compass Collatz Functions

\[ f(x, y) = (x - 1, y - 2) \]

\[ f(x, y) = (x - 1, y + 4) \]

\[ f(x, y) = (x + 1, y - 9) \]

\[ f(x, y) = (x + 1, y + 3) \]
A mortality problem for Compass Collatz-like Functions is shown \( \Pi^0_2 \)-complete, by reduction from mortality of 2-counter machines.

This problem is reduced to mortality of monic piecewise-affine functions on \( \mathbb{Z}^2 \).
Undecidability with a constant number of zones

We proved undecidability by reducing from Generalized Collatz functions, which in turn represent counter machines. The number of regions in the dynamical system is related to the size of the counter machine. For any constant $c$, mortality of counter machines smaller than $c$ is a decidable problem! The undecidability proof uses a sort of enhanced counter machine. This technique does not extend to monic functions.
We proved undecidability by reducing from Generalized Collatz functions, which in turn represent counter machines.

The number of regions in the dynamical system is related to the size of the counter machine.
We proved undecidability by reducing from Generalized Collatz functions, which in turn represent counter machines.

The number of regions in the dynamical system is related to the size of the counter machine.

For any constant $c$, mortality of counter machines smaller than $c$ is a decidable problem!
Undecidability with a constant number of zones

- We proved undecidability by reducing from Generalized Collatz functions, which in turn represent counter machines.
- The number of regions in the dynamical system is related to the size of the counter machine.
- For any constant \( c \), mortality of counter machines smaller than \( c \) is a decidable problem!
- The undecidability proof uses a sort of enhanced counter machine.
- This technique does not extend to monic functions.
Loops with a convex guard

In Program Termination analysis, loops are often of the form

$$\text{while } x \in G \text{ do } x \leftarrow f(x)$$

$G$ is a convex polyhedron $\{x \mid Ax \leq b\}$

Tiwari 2004, Braverman 2006 and some related works assumed $f$ to be affine. Decidability of termination of such loops over $\mathbb{Z}_n$ is still open (some special cases have been settled). The current work proves undecidability on $\mathbb{Z}_2$ when $f$ is piecewise affine with some (big enough) constant number of regions inside $G$. [BGM 2012] proves undecidability for two regions inside $G$ over $\mathbb{Z}_n$, where $n$ is unbounded. By combining the techniques, we can prove it for two regions inside $G$, with $n$ bounded (but big).

For what parameters is the problem decidable?
Loops with a convex guard

In Program Termination analysis, loops are often of the form

\[
\text{while } x \in G \text{ do } x \leftarrow f(x) \quad G \text{ is a convex polyhedron } \{ x \mid Ax \leq b \}
\]

[Tiwari 2004], [Braverman 2006] and some related works assumed \( f \) to be affine.

Decidability of termination of such loops over \( \mathbb{Z}^n \) is still open (some special cases have been settled).
Loops with a convex guard

In Program Termination analysis, loops are often of the form

while $x \in G$ do
  $x \leftarrow f(x)$

$G$ is a convex polyhedron $\{x \mid Ax \leq b\}$

[Tiwari 2004], [Braverman 2006] and some related works assumed $f$ to be affine.

Decidability of termination of such loops over $\mathbb{Z}^n$ is still open (some special cases have been settled).

The current work proves undecidability on $\mathbb{Z}^2$ when $f$ is piecewise affine with some (big enough) constant number of regions inside $G$.

[BGM 2012] proves undecidability for two regions inside $G$ over $\mathbb{Z}^n$, where $n$ is unbounded.

By combining the techniques, we can prove it for two regions inside $G$, with $n$ bounded (but big).
Loops with a convex guard

In Program Termination analysis, loops are often of the form

\[
\text{while } x \in G \text{ do } x \leftarrow f(x)
\]

\(G\) is a convex polyhedron \(\{x \mid Ax \leq b\}\)

[Tiwari 2004], [Braverman 2006] and some related works assumed \(f\) to be affine.

Decidability of termination of such loops over \(\mathbb{Z}^n\) is still open (some special cases have been settled).

The current work proves undecidability on \(\mathbb{Z}^2\) when \(f\) is piecewise affine with some (big enough) constant number of regions inside \(G\).

[BGM 2012] proves undecidability for two regions inside \(G\) over \(\mathbb{Z}^n\), where \(n\) is unbounded.

By combining the techniques, we can prove it for two regions inside \(G\), with \(n\) bounded (but big).

For what parameters is the problem decidable?