Computing to the infinite with ordinary differential equations.

Olivier Bournez[1] Sabrina Ouazzani[2]

[1]Ecole Polytechnique
Laboratoire d'Informatique de l'X

[2]Paris-Est Créteil University
LACL

March 2018
Algorithmic Questions In Dynamical Systems

Menu

$$y' = f(y)$$

Most classical frameworks:

- f is at least Lipschitz or $\mathcal{C}^1$ and computable;
- Cauchy-Lipschitz Theorem: existence and unicity of solutions;
- $\rightsquigarrow$ Turing machine simulation and vice-versa.

$$y' = f(y)$$

Most classical frameworks:

- f is at least Lipschitz or $\mathcal{C}^1$ and computable;
- Cauchy-Lipschitz Theorem: existence and unicity of solutions;
- $\rightsquigarrow$ Turing machine simulation and vice-versa.

Extended framework:

- what happens if f is (only) continuous?
- Cauchy-Peano-Ascoli Theorem: existence but non unicity of solutions;

$$y' = f(y)$$

Most classical frameworks:

- f is at least Lipschitz or $\mathcal{C}^1$ and computable;
- Cauchy-Lipschitz Theorem: existence and unicity of solutions;
- $\rightsquigarrow$ Turing machine simulation and vice-versa.

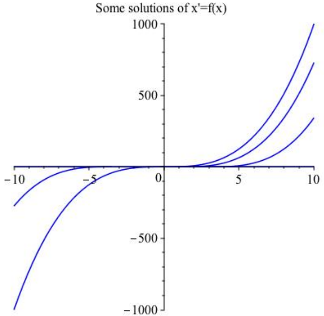
Extended framework:

- what happens if f is (only) continuous?
- Cauchy-Peano-Ascoli Theorem: existence but non unicity of solutions;
- $\rightsquigarrow$?

Non unicity: a classical counter-example

Solutions over $\mathbb{R}$ of f continuous such that:

$$\begin{cases} x' = f(x) \\ x(0) = 0 \end{cases} \quad \text{with} \quad \begin{cases} f(x) = 3x^{2/3} \\ f(0) = 0 \end{cases} \quad :$$



all functions $y_{-a,b}$ with $a, b \in \mathbb{R}^+ \cup \{+\infty\}$, where

$$y_{a,b}(t) = \begin{cases} 0 & \text{if } -a \leq t \leq b \\ (t+a)^3 & \text{if } t < -a \\ (t-b)^3 & \text{if } t > b \end{cases}$$

$$y' = f(y)$$

Most classical frameworks:

- f is at least Lipschitz or $\mathcal{C}^1$ and computable;
- Cauchy-Lipschitz Theorem: existence and unicity of solutions;
- $\rightsquigarrow$ Turing machine simulation and vice-versa.

Extended framework:

- what happens if f is (only) continuous?
- Cauchy-Peano-Ascoli Theorem: existence but non unicity of solutions;
- $\rightsquigarrow$?

- Main idea: describing transfinite time computations by (continuous time) dynamical systems and conversely.

- Concrete work: show that

> Continuous ordinary differential equations $\equiv$ Infinite time Turing machines.

Motivations:

- Applying gaps properties to Analysis.
- Applying the differential equation description to transfinite computation model.

Main result

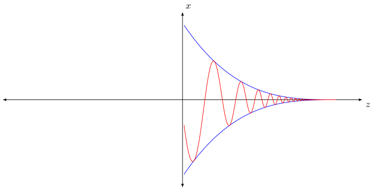
Theorem (ITTM are equivalent to $\mathcal{C}_0$-ODE's)

Any Infinite Time Turing Machine can be simulated by some continuous ordinary differential equation forward unique and vice-versa.

The idea

We consider $\mathcal{C}_0$-ODE, equations $x' = f(x)$, where

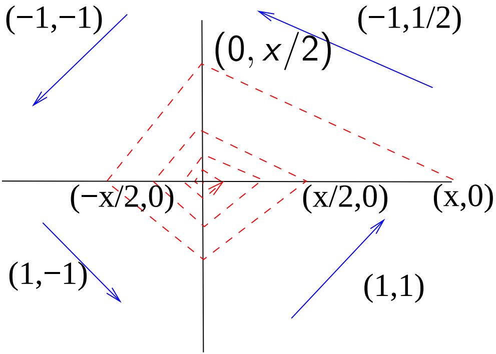
- $f : \mathbb{R}^n \to \mathbb{R}^n$;
- f is continuous.



Pedagogical illustration:

- f piecewise constant.

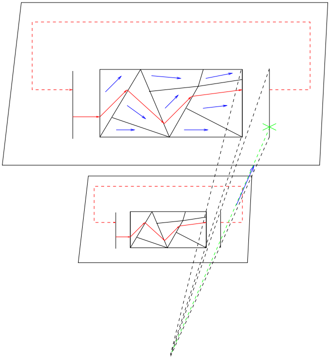
A trajectory of a Piecewise Constant Derivative System



$(0, 0)$ reached in:

- ω steps;
- in finite time $5/2(x + x/2 + x/4 + \dots) = 5x$.

Recognizing the halting problem of a Turing machine in dimension 4



Consequence: an ordinal time computation can be simulated in finite time.

PCD in continuous time? [Bournez99]

Extending [Asarin-Maler95].

Dimension	Languages semi-recognized
2	$< \Sigma_1$
3	Σ_1
4	Σ_2
5	Σ_ω
6	$\Sigma_{\omega+1}$
7	Σ_{ω^2}
8	Σ_{ω^2+1}
...	...
2p+1	$\Sigma_{\omega^{p-1}}$
2p+2	$\Sigma_{\omega^{p-1}+1}$

ITTM computational power

What about recognizing ITTMs in dimension at least 5?

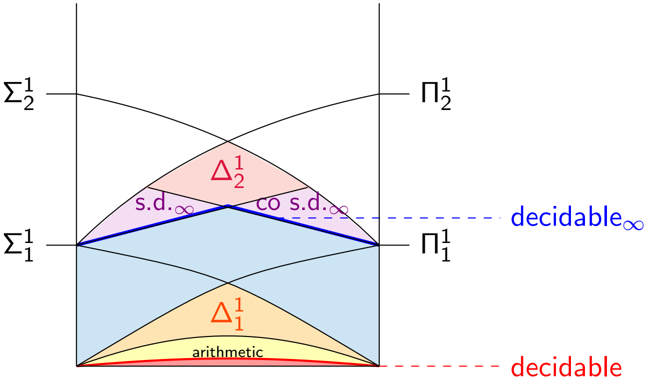


Figure: Projective hierarchy

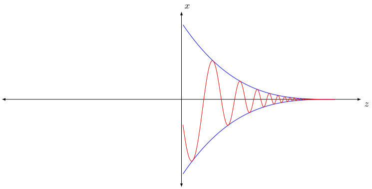
Here we have talked about PCD $\cdots$ what about $x' = f(x)$ with f continuous?

We want:

- to use continuous functions
- to compute using ordinals

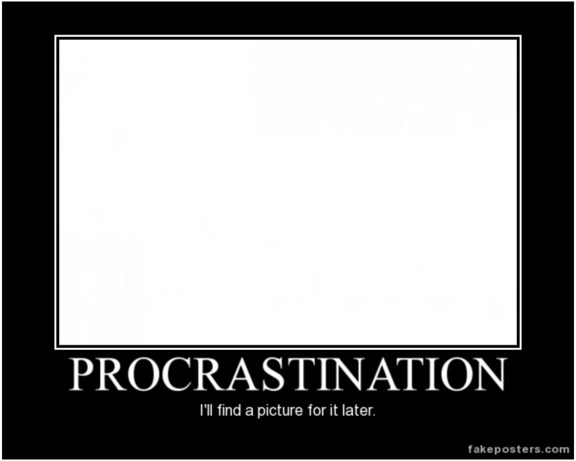
Accelerate an everywhere continuous dynamic.

- But still with f non-smooth on the whole space.
- Main construction: "Petard"



But for $x' = f(x)$ with f continuous, solutions exist necessarily but may be non unique.

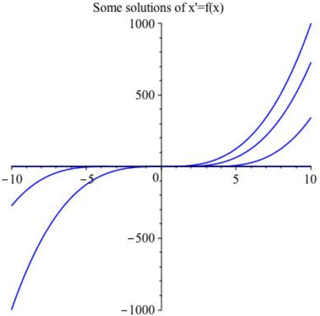
Non unicity and procrastination



Non unicity and procrastination: an illustration

Solutions over $\mathbb{R}$ of f continuous such that:

$$\left\{ \begin{array}{l} x' = f(x) \\ x(0) = 0 \end{array} \right. \quad \text{with} \quad \left\{ \begin{array}{l} f(x) = 3x^{2/3} \\ f(0) = 0 \end{array} \right. :$$



all functions $y_{-a,b}$ with $a, b \in \mathbb{R}^+ \cup \{+\infty\}$, where

$$y_{a,b}(t) = \left\{ \begin{array}{ll} 0 & \text{if } -a \leq t \leq b \\ (t+a)^3 & \text{if } t < -a \\ (t-b)^3 & \text{if } t > b \end{array} \right.$$

Computing with ordinals

We restrict to:

- non-procrastinating trajectories:
  - Basic observation: There is a trajectory from $x(0) = x_0$ to some point x^* iff there is some non-procrastinating trajectory from $x(0) = x_0$ to x^*.
- forward-unique:
  - locally unicity holds in terms of future for non-procrastinating trajectories

We want:

- to use continuous functions
- to compute using ordinals

Ordinals

Definition (Ordinal)

Transitive well-ordered set for the membership relation.

$$0 := \emptyset$$
$$1 := \{0\} = \{\emptyset\}$$
$$\cdots$$
$$\omega := \{0, 1, 2, 3, \cdots\}$$
$$\omega + 1 := \{0, 1, 2, 3, \cdots, \omega\}$$
$$\cdots$$
$$\omega.2 := \{0, 1, 2, \cdots, \omega, \omega{+}1, \omega{+}2 \ldots\}$$

- If α is an ordinal, then $\alpha \cup \{\alpha\}$, denoted $\alpha + 1$ is called successor of α and is an ordinal;
- let A be a set of ordinal numbers, then $\alpha = \bigcup_{\beta \in A} \beta$ is a limit ordinal.

Transfinite time computation models

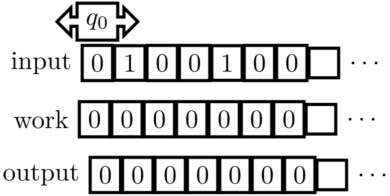
- Ordinals as time for computation.
- Peculiar ordinal properties.
- Different computation models: register machines, Turing machines, ordinal-tape Turing machines ...
- Proof of mathematical properties from an algorithmic point of view.

Structure of infinite time Turing machines (ITTM)

- 3 right-infinite tapes
- a single head
- binary alphabet $\{0, 1\}$

- additional special limit state lim
- computation steps are indexed by ordinals

Configuration

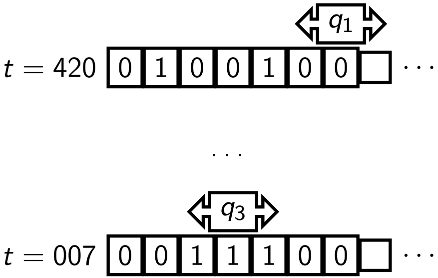


Operating an ITTM

Configuration at $\alpha + 1$.

$\rightsquigarrow$

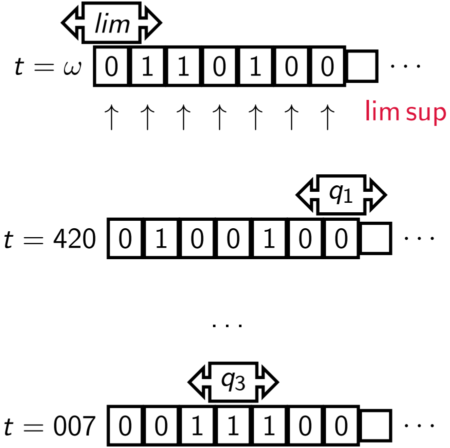
Configuration at α.



Operating an ITTM

Configuration limit:

- head: initial position;
- state: lim;
- each cell: *lim sup* of cell values before.



Some litterature

- Machines halt when they reach the halting state.
- Either an ITTM halts in a countable numer of steps, either it begins looping in a **countable number of steps** ([HL00]).
- *WO* is decidable by an ITTM.
- Every Π^1_1 and Σ^1_1 set is decidable by an ITTM.

Peculiar ITTM ordinals

Halting on input $000\ldots \rightsquigarrow$ two natural notions of ordinals.

Definition (Clockable ordinal)

α clockable: there exists an ITTM that **halts** on input $000\ldots$ in exactly α steps of computation.

Definition (Writable ordinal)

α writable: there exists an ITTM that **writes a code** for α on input $000\ldots$ and halts.

Theorem (Welch [Wel09])

The supremum of the clockable ordinals is equal to the supremum of the writable ordinals. It is called λ.

λ is a rather large countable recursively inacessible ordinal. . .

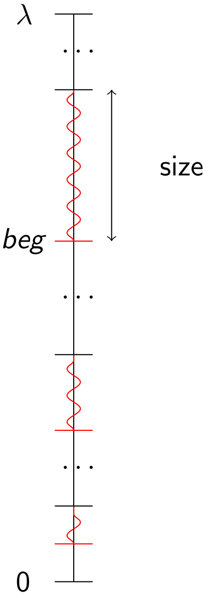
Gap

There exist writable ordinals that are not clockable such that:

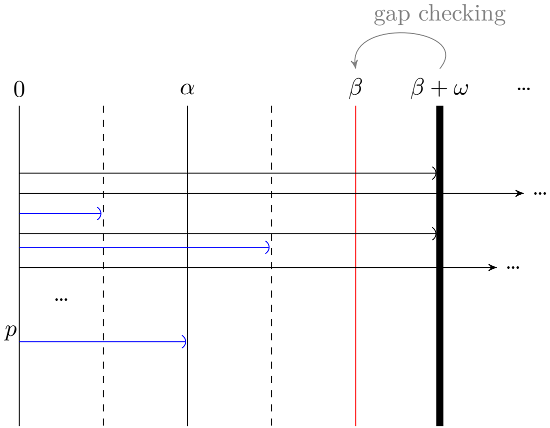
- they form intervals;
- these intervals have limit sizes.

Definition (Gap)

Intervals of not clockable ordinals.

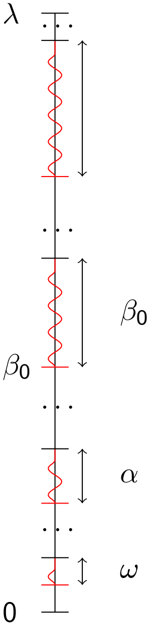


Proof of gap existence



Simulation of all programs on input 0.

In blue: halting programs. In red: limit step, begins a gap?



gaps containing admissible ordinals

admissible = beginning of gaps

We want:

- to use continuous functions
- to compute using ordinals

Menu

Simulation of an ITTM by a $\mathcal{C}_0$-ODE

- successor case: classical Turing machine
- limit case: accelerating the computation to compute the limit tape.

Simulation of an ITTM by a $\mathcal{C}_0$-ODE

- successor case: classical Turing machine
- limit case: accelerating the computation to compute the limit tape.

Successor case

- discrete time dynamical system;
- differential equations.

Successor case: Turing Machines

- Let M be some one tape Turing machine, with m states and 10 symbols.
- If

$$\ldots B\ B\ B\ a_{-k}\ a_{-k+1}\ldots\ a_{-1}\ a_0\ a_1\ldots\ a_n\ B\ B\ B\ldots$$

  is the tape content of M, it can be seen as

$$\begin{aligned} y_1 &= a_0 a_1 \ldots a_n \\ y_2 &= a_{-1} a_{-2} \ldots a_{-k} \end{aligned} \quad (1)$$

- The configuration of M is then given by three values: its internal state s, y_1 and y_2.

Successor case: Alternative view of a Turing machine

$$
\begin{aligned}
y_1 &= a_0 10^{-1} + a_1 10^{-2} + \ldots + a_n 10^{-n-1} \\
y_2 &= a_{-1} 10^{-1} + a_{-2} 10^{-2} + \ldots + a_{-k} 10^{-k}.
\end{aligned} \tag{2}
$$

Turing Machine	PAM
State Space $\{q_1, q_2, \cdots, q_m\} \times \Sigma^*$	State Space $[1, m+1] \times [0, 1]$
State $(q_i, a_{-m} \ldots a_{-1}, a_0 \ldots a_n)$	State $x = s + y_2$
q_1: if 2 is read, then write 4; goto q_2	$\begin{cases} x := x + 1 \\ y_1 := y_1 + \frac{2}{10} \end{cases}$ if $\begin{cases} 1 \leq x < 2 \\ \frac{2}{10} \leq y_1 < \frac{3}{10} \end{cases}$
q_5: if 3 is read, then move right; goto q_1	$\begin{cases} x := \frac{x-5}{10} + \frac{3}{10} + 1 \\ y := 10 * y - 3 \end{cases}$ if $\begin{cases} 5 \leq x < 6 \\ \frac{3}{10} \leq y < \frac{4}{10} \end{cases}$
q_3: if 5 is read, then move left; goto q_7	$\begin{cases} x := 10(x-3) - j + 7 \\ y := \frac{y}{10} + \frac{j}{10} \end{cases}$ if $\begin{cases} 3 + \frac{j}{10} \leq x < 3 + \frac{j+1}{10} \\ \frac{5}{10} \leq y_1 < \frac{6}{10} \end{cases}$ for $j \in \{0, 1, \ldots, 9\}$.

$f(t) = (x(t+1), y_1(t+1))$ piecewise affine $\rightsquigarrow$ can be made smooth.

.

Iterating a function with an ODE

$z_1(t) = (x(t), y(t))$

We want to alternate $z_2 := f(z_1)$, $z_1 := z_2$.

Iterating a function with an ODE

$z_1(t) = (x(t), y(t))$

We want to alternate $z_2 := f(z_1)$, $z_1 := z_2$.

$\rightsquigarrow$ Branicky's clock (1995).

The solution of $y' = c(g-y)^3\phi(t)$:

- converges at $t = 1/2$ the goal g;
- with some arbitrary precision;
- independantly from initial condition at $t = 0$;
- this roughly does $y(1/2) := g$.

The following system is a solution:

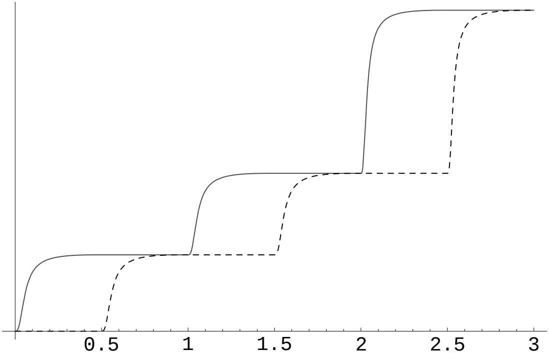
$$\begin{cases} z_1' = c_1(z_2 - z_1)^3\theta(-\sin(2\pi t)) \\ z_2' = c_2(f(z_1) - z_2)^3\theta(\sin(2\pi t)) \end{cases} \quad \begin{cases} z_1(0) = x_0 \\ z_2(0) = x_0 \end{cases}$$

considering functions:

- θ such that $\theta(x) = 0$ if $x \leq 0$;
- $\theta(x) = x^2$ if $x \geq 0$.

Illustrative example: $z_1(t+1) := 2 * z_1(t)$

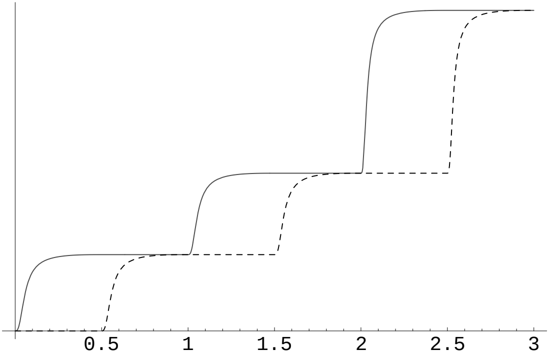
We want to alternate $z_2 := f(z_1)$, $z_1 := z_2$. Here $f = " \times 2"$.



Simulation of iterations of $h(n) = 2^n$ by ODEs.

Illustrative example: $z_1(t+1) := 2 * z_1(t)$

We want to alternate $z_2 := f(z_1)$, $z_1 := z_2$. Here $f =$ " $\times$ 2".



Simulation of iterations of $h(n) = 2^n$ by ODEs.

Any Turing machine can be simulated by iterating as above.

Simulation of an ITTM by a $\mathcal{C}_0$-ODE

- successor case: classical Turing machine
- limit case: accelerating the computation to compute the limit tape.

Limit case: Aim

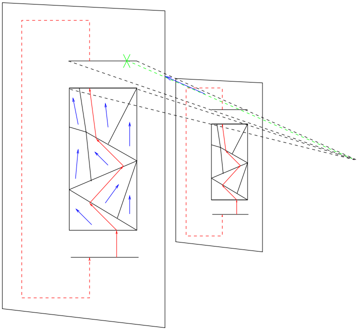
- compute the limit of a computation;
- send the result to the next successor simulation computation.

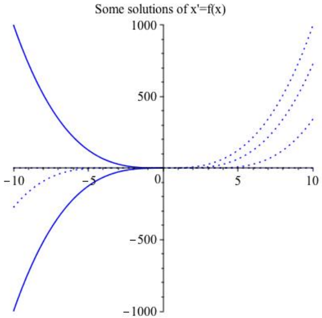
Limit case: Aim

- compute the limit of a computation;
- send the result to the next successor simulation computation.

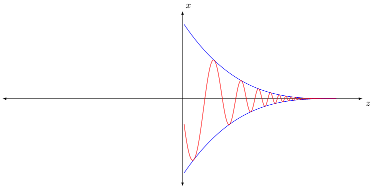
Continuous petard: a change of variables · · ·

Time + space





$\cdots$ to obtain an "accumulation"



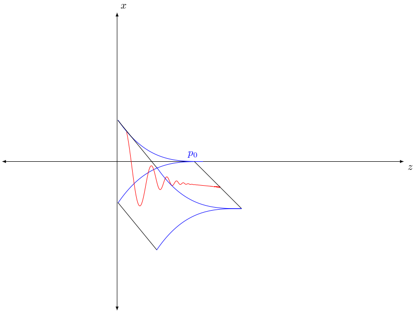
Build a $\mathcal{C}_0$-ODE such that its trajectories:

- start from $X(0) = (x, x_0)$;
- are simulating the previous ones;
- in a time bounded by 1.

Content of the limit tape

Limit step: content of the limit tape = computation of a serie;

- some of the variables go to 0;
- some others encode the value of the series.



Content of the limit tape: the limit convention

- Emulating a lim sup is not easy.
- Equivalent easier convention: at limit ordinal time ρ, if μ is the ordinal written in the ordinal tape then the content of a cell of limit tape is 1 (and 0 otherwise) iff it was already 1 at a time $\mu < \pi < \rho$.

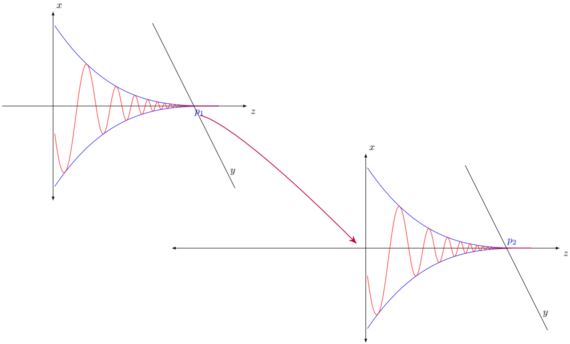
Limit on the encodings = limit on the tapes.

Limit case: aim

- compute the limit of a computation;
- send the result to the next successor simulation computation.

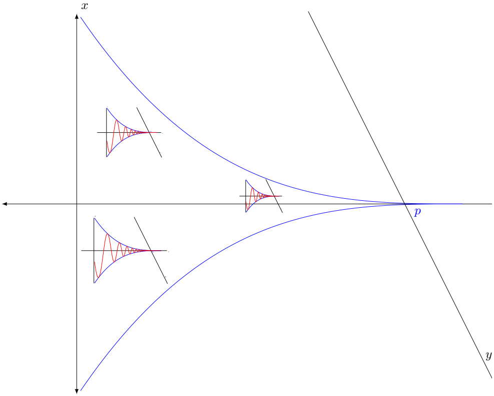
Path between two petards

- The result of the computation by the first petard, obtained by some real number p_1 (encodinga limit tape), is sent to the second.



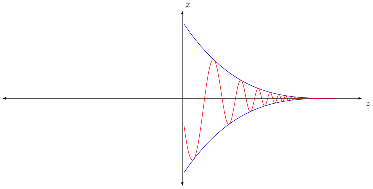
Nested petards

- Imbrication of petards in a fixed dimension space.
- Different from PCD where limits necessarily imply that dimension must increase.



Menu

Inversion of petards



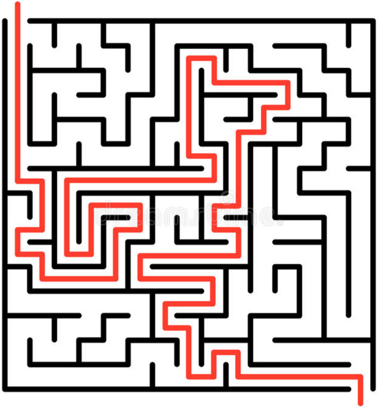
Idea: simulate the dynamics in the other direction.

But $\cdots$

We want an algorithm to solve $\mathcal{C}_0$-ODE, but:

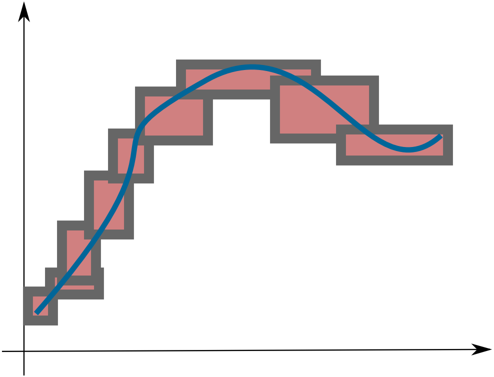
- non-unicity of the solutions;
- continuous and general equation.

Existential point of view



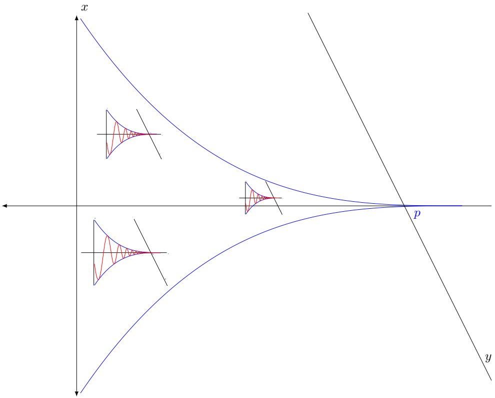
If a solution exists, ten thousand monkeys will finish to find it!

The ten thousand Monkeys algorithm



- ten thousand monkeys algorithm of [CG09];
- covering the solution by a finite amount of boxes.

The ω-Monkeys algorithm



- extended version to output solutions using ITTM;
- covering the solution by an ordinal amount of boxes.

Main result

Theorem (ITTM are equivalent to $\mathcal{C}_0$-ODE's)

Any Infinite Time Turing Machine can be simulated by some computable (hence continuous) ordinary differential equation forward uniqueand vice-versa.

Consequences

infinite time Turing machines

$=$

model for algorithms proving logical properties

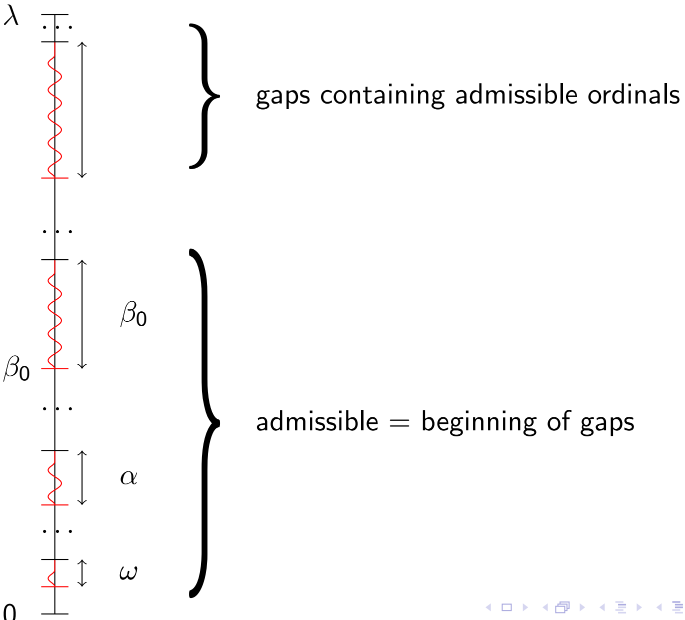
Continuous ordinary differential equations $\equiv$ Infinite time Turing machines.

Consequences

- Every non-procrastinating infinite time trajectory of a $\mathcal{C}_0$-ODE either halts or repeats itself in countably many steps.
- *WO* is decidable by some computable (hence continuous) ordinary differential equation.
- Every Π^1_1 set is decidable by some computable (hence continuous) ordinary differential equation. Hence, every Σ^1_1 set is decidable by some computable (hence continuous) ordinary differential equation.

Menu

Back to ITTM: gaps in computation times



Consequences and questions

- Are there gaps in the $\mathcal{C}_0$-ODE?
- Can we define only countable ordinals?

- Applying transfinite techniques to Analysis.
- Transposing Analysis questions to transfinite computations.
- 2 dual views for the same computability questions.
- discrete transfinite time $=$ continuous time.

Thank you for your attention.

Some references:

P. Collins and D.S. Graça.
Effective Computability of Solutions of Differential Inclusions The Ten Thousand Monkeys Approach.
Journal of Universal Computer Science, 15(6):1162–1185, 2009.

Joel D. Hamkins and Andrew Lewis.
Infinite time turing machines.
Journal of Symbolic Logic, 65(2):567–604, 2000.

Philip D. Welch.
Characteristics of discrete transfinite time turing machine models: Halting times, stabilization times, and normal form theorems.
Theoretical Computer Science, 410(4-5):426–442, 2009.

Encoding countable ordinals

Countable ordinal = well order on $\mathbb{N}$.
Encoding countable ordinals by reals:
Let $<$ be an order on the natural numbers.
The real r is a code for the order-type of $<$ if, for $i = \langle x, y \rangle$, the i-th bit of r is $\mathbf{1}$ if and only if $x < y$.
Example: $\omega.2 = \omega + \omega \rightsquigarrow$ even integers lower than odd integers.

$$0 = \langle 0, 0 \rangle \ 1 = \langle 0, 1 \rangle \ \cdots \ r = 0_0 1_1 0_2 0_3 0_4 1_5 0_6 1_7 1_8 1_9 1_{10} \cdots$$

Arithmetical hierarchy

- Σ_1 = Recursively enumerable sets.
- Σ_2 = Sets recursively enumerable in a set in Σ_1.
- Σ_3 = Sets recursively enumerable in a set in Σ_2.
- ...
- Σ_{k+1} = Sets recursively enumerable in a set in Σ_k.
- ...

Hyper-arithmetical hierarchy

- Σ_1 = Recursively enumerable sets.
- ...
- Σ_{k+1} = Sets recursively enumerable in a set in Σ_k.
- ...
- Σ_ω = Sets recursively enumerable in a diagonalisation of $\Sigma_{\gamma<\omega}$
- $\Sigma_{\omega+1}$ = Sets recursively enumerable in a set in Σ_ω
- ...
- $\Sigma_{\alpha=\lim\gamma}$ = Sets recursively enumerable in a diagonalisation of $\Sigma_{\gamma<\alpha}$.
- $\Sigma_{\alpha+1}$ = Sets recursively enumerable in a set of Σ_α
- ...