GPAC, Analyse récursive et fonctions $\mathbb{R}$-récursives
Trois modèles équivalents de calcul sur les réels.

Emmanuel Hainry

LORIA/INPL, Nancy, France

Discrete Case

- There are several models for computation over integers
  - Recursive functions
  - Turing machines
  - Circuits
  - λ-calculus
  - ...
- But those models are "equivalent".

Church-Turing thesis

All reasonable powerful enough discrete models of computation compute exactly the same functions.

Approaches to analog computation

Several different devices

- Differential analyzer [Bush 31]
- Neural networks [Hopefield 84]
- Operational Amplifiers
- ...

Several different models:

- General Purpose Analog Computer (GPAC) [Shannon 41]
- Computable Analysis [Turing 36]
- BSS model [Blum Shub Smale 89]
- ...

However, contrarily to the digital case, few connections between these models are known.

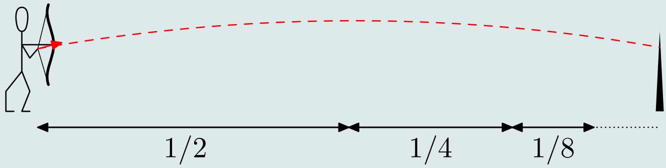
Linking models of “real” computation

- The models of computable analysis and $\mathbb{R}$-recursive functions deal with similar functions but lack relations between their classes.
- Investigating such links can help giving an analog characterization of what may be considered reasonable in computation over the reals.
- A step towards a Church Thesis for computation over the reals?
- A way to characterize the algorithmic complexity of some problems on dynamical systems?

“Real-time” computing yields unreasonable results

Zeno paradox (Ζήνον ὁ Ελεάτης)

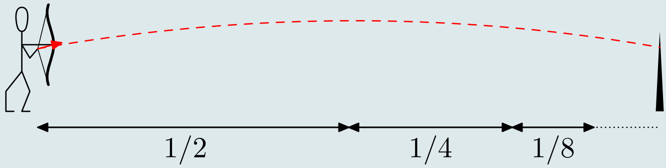
At any time between its launch and arrival, an arrow has first to cover one half of the distance towards its goal.



“Real-time” computing yields unreasonable results

Zeno paradox (Ζήνον ὁ Ελεάτης)

At any time between its launch and arrival, an arrow has first to cover one half of the distance towards its goal.

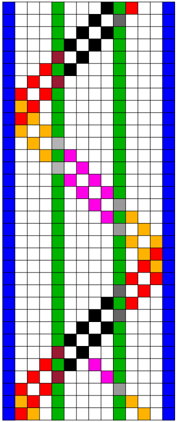


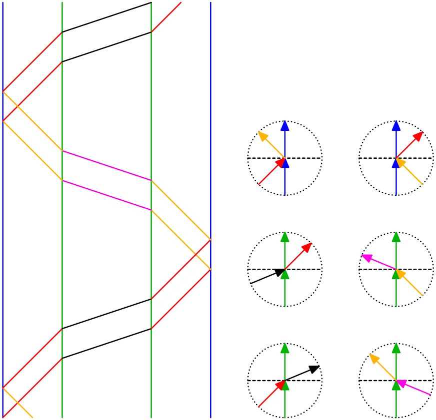
Accelerating Turing machine

An ATM achieves its first computing step in time $\frac{1}{2}$, its second step in time $\frac{1}{4}$, its n-th step in time $\frac{1}{2^n}$.
At time 1, this machine has done an infinity of computation steps.

Zeno phenomenon in signal machines

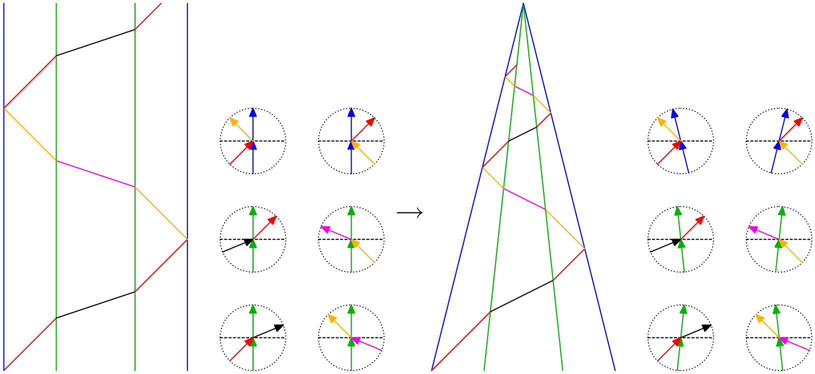
Signal machines [Durand-Lose 03] are a continuous counterpart to cellular automata:





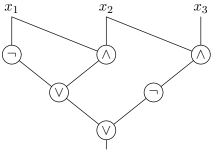
Zeno phenomenon in signal machines (2)

It is possible to reduce the time taken to do a computation by changing the slopes of the signals:

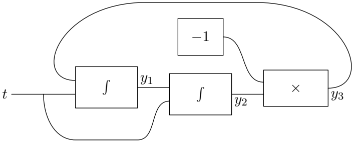


Setting

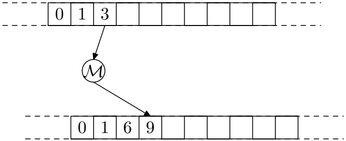
Circuit



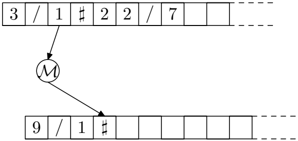
GPAC



Turing Machine:



Recursive analysis:



Recursive functions:

$[0, S, U; COMP, REC, \mu]$

$\mathbb{R}$-recursive functions:

$[0, 1, U; \mathrm{COMP}, \mathrm{INT}, \mu_{\mathbb{R}}]$

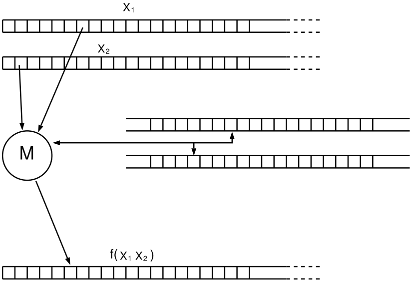
Recursive and Sub-recursive functions

$$\begin{array}{rcl}
\mathcal{R}ec(\mathbb{N}) & = & [0, S, U; COMP, REC, \mu] \\
\cup\!\!\not\,\!\shortmid & & \\
\mathcal{PR}(\mathbb{N}) & = & [0, S, U; COMP, REC] \\
\cup\!\!\not\,\!\shortmid & & \\
\mathcal{E}_n(\mathbb{N}) & = & [0, S, U, \ominus, E_{n-1}; COMP, B\Sigma, B\Pi] \\
\cup\!\!\not\,\!\shortmid & & \\
\mathcal{E}_3(\mathbb{N}) = \mathcal{E}(\mathbb{N}) & = & [0, S, U, \ominus; COMP, B\Sigma, B\Pi]
\end{array}$$

Recursive and Sub-recursive functions

$$\begin{array}{ccl}
\mathcal{R}ec(\mathbb{N}) & \sim & \text{Turing machines} \\
\cup\!\!\!\not| & & \\
\mathcal{PR}(\mathbb{N}) & \sim & \text{For programs (no while)} \\
\cup\!\!\!\not| & & \\
\mathcal{E}_n(\mathbb{N}) & & \text{Grzegorczyk's hierarchy} \\
\cup\!\!\!\not| & & \\
\mathcal{E}_3(\mathbb{N}) = \mathcal{E}(\mathbb{N}) & \sim & \text{Time bounded by a } 2^{2^{2^{\cdots^{n}}}}
\end{array}$$

Recursive analysis: type 2 machines



A tape represents a real number

Let $\nu_{\mathbb{Q}}$ be a representation of the rational numbers.
$(x_n) \rightsquigarrow x$ iff $\forall i, |x - \nu_{\mathbb{Q}}(x_i)| < \frac{1}{2^i}$

M behaves like a Turing machine

Write-only one-way output tape.

Computable functions

Definition [Computable functions]

A function $f : [a, b] \to \mathbb{R}$ with $a, b \in \mathbb{Q}$ is computable (resp: elementarily computable) iff there exists $\phi : \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}}$ recursive (resp: elementary) such that

$$\forall X \rightsquigarrow x, (\phi(X)) \rightsquigarrow f(x).$$

Examples of recursively computable functions

Most usual functions are recursively computable:

- Polynomials, exp, sin, cos are in $\mathcal{R}ec(\mathbb{R})$
- Euler's Γ

  $$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt$$

  is in $\mathcal{R}ec(\mathbb{R})$

All functions defined through recursive analysis are continuous.

Differential analyzers/GPAC

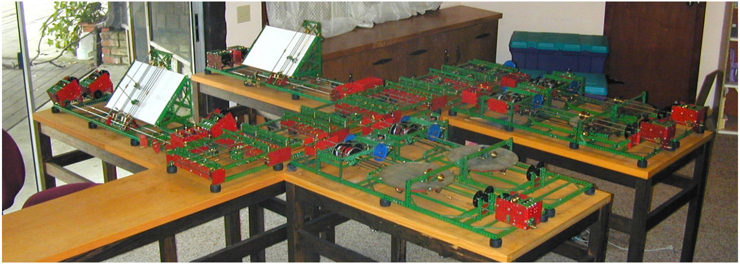
1876 William Thomson (Lord Kelvin) first thought of interconnecting mechanical integrators to compute.

1931 A differential was built by Vannevar Bush at MIT.

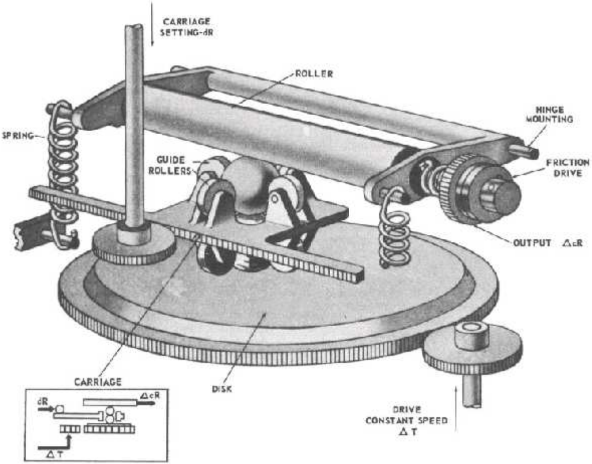
1941 Claude Shannon modelized the differential analyzer as a GPAC.

Differential analyzer

http://www.meccano.us/differential_analyzers/robinson_da/

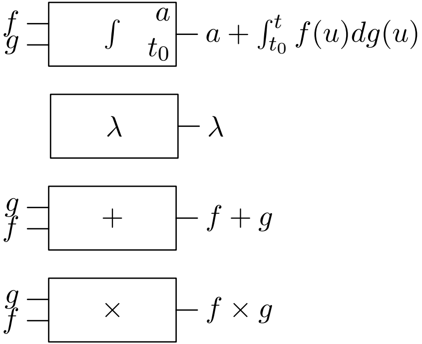


Mechanical integrator



The GPAC

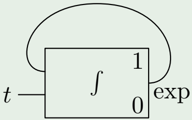
GPAC [Shannon 41] consists in circuits interconnecting the following components:



There can be loops in the circuit.

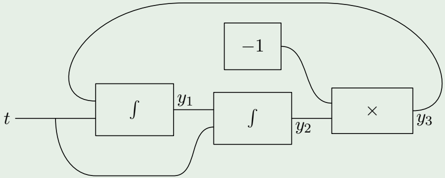
Examples

Example (Computing exp with a GPAC)



$$\begin{cases} y' = y \\ y(0) = 1 \end{cases}$$

Example (Computing cos and sin with a GPAC)



$$\begin{cases} y_1(0) = 1 \\ y_2(0) = 0 \\ y_2' = y1 \\ y_1' = -y_2 \end{cases}$$
$$\Longrightarrow \begin{cases} y_1 = \sin \\ y_2 = \cos \end{cases}$$

Features of the GPAC

Claim [Shannon 41]

Functions generated by GPAC are the differentially algebraic functions.
Differentially algebraic functions are the solutions of $P\left(t, y, y', ..., y^{(n)}\right) = 0$.

The proof was corrected in [Pour-El 74] then [Lipshitz Rubel 87] and [Graça Costa 03].

Theorem [Graça Costa 03]

A scalar function $f : \mathbb{R} \to \mathbb{R}$ is generated by a GPAC iff it is a component of the solution of a system

$$y' = p(t, y), \tag{1}$$

where p is a vector of polynomials.

Previous results on the GPAC

It can be shown that:

- The GPAC computes most usual functions (polynomials, trigonometric functions...)
- The Gamma function $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt$ and Riemann's zeta function $\zeta(x) = \sum_{n=1}^\infty \frac{1}{n^x}$ cannot be computed by a GPAC

The latter result seems to indicate that the GPAC is less powerful than recursive analysis since Γ and ζ are computable according to Computable Analysis.

$\mathbb{R}$-recursive functions [Moore 96]

Definition [$\mathcal{G}$]

$\mathcal{G} = [0, 1, U; \mathrm{COMP}, \mathrm{INT}, \mu_{\mathbb{R}}]$

$$\begin{array}{c} REC : f, g \mapsto h \\ h(x, 0) = f(x) \\ h(x, S(n)) = g(x, n, h(x, n)) \end{array} \qquad \begin{array}{c} \mathrm{INT} : f, g \mapsto h \\ h(x, 0) = f(x) \\ \frac{\partial h}{\partial y}(x, y) = g(x, y, h(x, y)) \end{array}$$

Problems with $\mathcal{G}$

- Not always well defined ($0 \times +\infty = 0$, non integrable functions).
  - [Mycka Costa 04] presents well-defined operator (differential recursion and infinite limits) that have the same power as $\mathcal{G}$.
- Presents time compression phenomenon (Zeno's paradox).
- Contains unwanted functions (in particular $\chi_{\mathbb{Q}}$ or functions that decide the halting problem of Turing machines).

$\chi_{halt} \in \mathcal{G}$

- There exists $f : \mathbb{R}^3 \to \mathbb{R}^3$ simulating a Turing machine: $\forall m, n, q \in \mathbb{N}^3$, $(m', n', q') = f(m, n, q)$ is the next configuration (m, n represent the tape, q the state).
- Iteration can be simulated in $\mathcal{G}$. $F(t, m_0, n_0, q_0)$ represents the configuration after t steps.
- $\mathcal{M}$ halts iff $\exists t \in \mathbb{N}$ such that $(\_, \_, q_f) = F(t, m_0, n_0, q_0)$
- in other terms, $\mathcal{M}$ if and only if the smallest root of

$$(u_3(F(tan(z), ...)) - q_f)(z - \pi/2)$$

is not $\pi/2$.

Setting

	GPAC
Turing machines	Recursive analysis
Recursive functions	$\mathbb{R}$-recursive functions

We have seen that Γ belongs to $\mathcal{R}ec(\mathbb{R})$ but is not generable by GPAC.

GPAC with limit

The notions of computability in the GPAC and in Computable Analysis are very distinct: "real time" computation *versus* limit procedure

Definition

1. Use initial settings on integrators to represent the initial input $x \in \mathbb{R}^n$ (the other initial settings must be computable reals).
2. Use the usual input as a time variable t
3. Then $f : \mathbb{R}^n \to \mathbb{R}$ is GPAC-computable if there is a GPAC with two outputs $g(x,t)$ and $\varepsilon(x,t)$ satisfying:
   - $\lim_{t\to\infty} \|\varepsilon(x,t)\| = 0$;
   - $\|g(x,t) - f(x)\| \leq \varepsilon(x,t)$

How to compute Γ with a GPAC

- This notion can be expected to match more closely Computable Analysis.
- In [Graça 04] it is shown that Γ and ζ are GPAC-computable.
- But no exact characterization of the class of functions obtained by the previous notion was previously given.

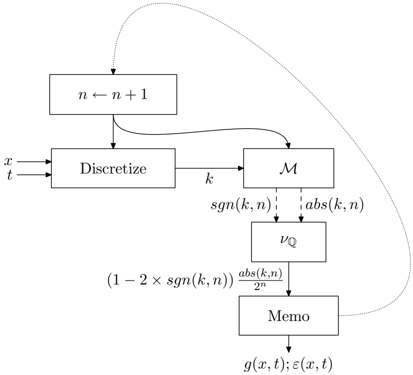
Result

Theorem [with Bournez Campagnolo Graça]

Let $f : [a, b] \to \mathbb{R}$ be a real function. Then f is recursively computable iff it is GPAC-computable.

recursively computable $\Rightarrow$ GPAC-computable

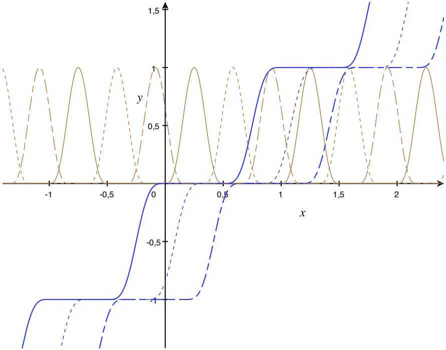
We use results from [Branicky 95] and [Graça Campagnolo Buescu 05] to build a GPAC that simulates robustly a Turing machine.



1. Compute an integer k from x and n such that $\left|k/2^{m(n)} - x\right| < 1/2^{m(n)}$;
2. Compute $sgn(k,n)$ and $abs(k,n)$;
3. Compute $\frac{(1-2sgn(k,n))abs(k,n)}{2^n}$ and memorize the result till another cycle is completed;
4. Take $n = n+1$ and restart the cycle.

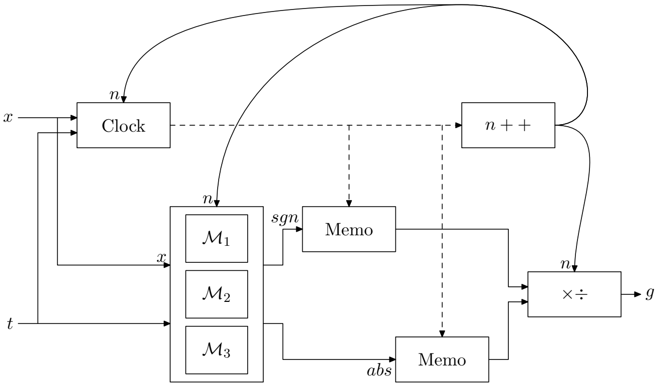
Simulating the discrete part

- We would like to take $k = \lfloor x2^{m(n)} \rfloor$, but the discrete function "integer part" $\lfloor \cdot \rfloor$ cannot be obtained by a GPAC
- Our solution is to use three functions $r_i(t)$ and three "detecting functions" ω_i such that $\omega_i(t) \neq 0$ iff $r_i(t) \in \mathbb{N}$



$$y = \frac{\sum_{i=0}^{2} \omega_{k,i}(nx) s_i}{\sum_{i=0}^{2} \omega_{k,i}(nx)}$$

Recursively computable implies GPAC-computable



Setting

	Recursive analysis
Turing machines	Type-2 machines
Recursive functions	$\mathbb{R}$-recursive functions
	$\mathcal{G}$, $\mathcal{L}$, $\mathcal{H}$

We know that $DP(\mathcal{L}) = \mathcal{E}(\mathbb{N})$.
To characterize in an algebraic way the real recursive functions, we will define a zero-finding operator and a limit operator.

$\mathbb{R}$-recursive functions [Campagnolo Moore Costa 00]

Definition [$\mathcal{L}$]

$$\mathcal{L} = [0, 1, -1, \pi, U, \theta_3; \mathrm{COMP}, \mathrm{LI}]$$

With

- U : projections
- $\theta_3 : \begin{cases} \mathbb{R} & \longrightarrow & \mathbb{R} \\ x & \mapsto & \max(0, x^3) \end{cases}$
- COMP: composition
- LI: given g, h. $f = \mathrm{LI}(g, h)$ is the maximal solution of

$$\begin{aligned} f(\overrightarrow{x}, 0) &= g(\overrightarrow{x}) \\ \frac{\partial f}{\partial y}(\overrightarrow{x}, y) &= h(\overrightarrow{x}, y) f(\overrightarrow{x}, y) \end{aligned}$$

Properties of $\mathcal{L}$

For a class $\mathcal{F}$ of functions $\mathbb{R} \to \mathbb{R}$, $DP(\mathcal{F})$ is the set of functions $\mathbb{N} \to \mathbb{N}$ that have an extension in $\mathcal{F}$.

Theorem [Campagnolo Moore Costa 00]

$$DP(\mathcal{L}) = \mathcal{E}(\mathbb{N})$$

Theorem [Campagnolo Moore Costa 00]

$$DP(\mathcal{L}_n) = \mathcal{E}_n(\mathbb{N})$$

Extension to recursive functions

- This result gives a characterization of $\mathcal{E}(\mathbb{N})$ (and has been extended to all levels of the Grzegorczyk hierarchy).
- We introduce an operator UMU to obtain

$$DP(\mathcal{L} + \mathrm{UMU}) = \mathcal{R}ec(\mathbb{N}).$$

A real μ operator

Remark: A naive "return the smallest root" operator yields unwanted functions (see [Moore 96]).

Definition

Given $f : \mathcal{D} \times \mathcal{I} \subset \mathbb{R}^{k+1} \to \mathbb{R}$ differentiable such that:

- $\forall \overrightarrow{x} \in \mathcal{D}$, the function $g_{\overrightarrow{x}} : y \mapsto f(\overrightarrow{x}, y)$ is non decreasing,
- $g_{\overrightarrow{x}}$ has a unique root $y_{\overrightarrow{x}} \in \mathring{\mathcal{I}}$,
- $\frac{\partial f}{\partial y}(\overrightarrow{x}, y_{\overrightarrow{x}}) > 0$.

$$\mathrm{UMU}(f) = \begin{cases} \mathbb{R}^k & \longrightarrow & \mathbb{R} \\ \overrightarrow{x} & \mapsto & y \text{ such that } f(\overrightarrow{x}, y) = 0 \end{cases}$$

$\mathcal{H} = \mathcal{L} + \mathrm{UMU}$

Definition [$\mathcal{H}$]

$$\mathcal{H} = [0, 1, U, \theta_3; \mathrm{COMP}, \mathrm{CLI}, \mathrm{UMU}]$$

Proposition

$$\mathcal{H} = \mathcal{L} + \mathrm{UMU}$$

Proof:

- $-1 = \mathrm{UMU}(x \mapsto x + 1)$
- $x \mapsto \frac{1}{1+x^2} = \mathrm{UMU}\left(x, y \mapsto (1 + x^2)y - 1\right)$;
  $\arctan(0) = 0$ and $\arctan'(x) = \frac{1}{1+x^2}$;
  $\pi = 4 \arctan(1)$

Result: $DP(\mathcal{H}) = \mathcal{R}ec(\mathbb{N})$

Theorem

$$DP(\mathcal{H}) = \mathcal{R}ec(\mathbb{N})$$

Where $\mathcal{R}ec(\mathbb{N})$ denotes the set of discrete partial recursive functions.

Proof: we have to demonstrate both directions.

- $DP(\mathcal{H}) \subset \mathcal{R}ec(\mathbb{N})$ comes from the fact that UMU preserves computability (in the sense of recursive analysis).
- $\mathcal{R}ec(\mathbb{N}) \subset DP(\mathcal{H})$ can be proven using a normal form theorem in $\mathcal{R}ec(\mathbb{N})$ and translating the discrete μ into our UMU.

Consequences

Corollary

$$\mathcal{L} \subsetneq \mathcal{H}$$

Theorem [Normal Form]

A function from $\mathcal{H}$ can be written with at most 3 nested UMU.

We may need 2 UMU to obtain π and -1. The other UMU comes from the simulation of the discrete μ.

Characterizing computable analysis classes

- Previous results give analog characterizations of $\mathcal{E}(\mathbb{N})$ and $\mathcal{R}ec(\mathbb{N})$.
- With a limit operator, we can extend those characterizations to obtain characterizations of $\mathcal{E}(\mathbb{R})$ and $\mathcal{R}ec(\mathbb{R})$.

$$\mathcal{H} + \mathrm{LIM} = \mathcal{R}ec(\mathbb{R})$$

- From [Mycka Costa 04], we know that a natural limit operator is as powerful as Moore's $\mu_{\mathbb{R}}$.

Operator LIM

Definition

Given $f : \mathbb{R} \times \mathcal{D} \subset \mathbb{R}^{k+1} \to \mathbb{R}^l$,

- if there are $K : \mathcal{D} \to \mathbb{R}$ and $\beta : \mathcal{D} \to \mathbb{R}$ *polynomials* such that

$$\forall \overrightarrow{x}, \forall t \geq \|\overrightarrow{x}\|, \|\frac{\partial f}{\partial t}(t, \overrightarrow{x})\| \leq K(\overrightarrow{x}) \exp(-t\beta(\overrightarrow{x})),$$

- if $\overrightarrow{x} \mapsto \lim_{t \to +\infty} f(t, \overrightarrow{x})$ is $\mathcal{C}^2$.

Then, $F = \mathrm{LIM}(f, K, \beta)$ is defined by

$$F(\overrightarrow{x}) = \lim_{t \to \infty} f(t, \overrightarrow{x}).$$

Theorems

We will write $\mathcal{C}^\star$ where $\mathcal{C} = [\mathcal{F}; \mathcal{O}]$ to denote the class $[\mathcal{F}; \mathcal{O}, \mathrm{LIM}]$.

Theorem

For functions of class $\mathcal{C}^2$ defined on a compact domain,

$$\mathcal{L}^\star = \mathcal{E}(\mathbb{R}).$$

Theorems

Theorem

For functions of class $\mathcal{C}^2$ defined on a compact domain,

$$\mathcal{H}^\star = \mathcal{R}ec(\mathbb{R}).$$

Consequences

Theorem [Normal Form]

A function from $\mathcal{L}^\star$ or $\mathcal{H}^\star$ can be written with at most 2 nested LIM

One limit to obtain $1/x$ and another from the limit mechanism.

Proposition

Let $\bar{D} = [0, 1, -1, U; \mathrm{COMP}, \bar{I}]$.

$$(\bar{D} + \theta_3)^* \supseteq \mathcal{PR}(\mathbb{R}).$$

Results

GPAC-computable $\Leftrightarrow$ Recursively computable

$$\mathcal{R}ec(\mathbb{R}) = \mathcal{H}^\star$$
$$\mathcal{PR}(\mathbb{R}) \subseteq (\bar{\mathcal{D}} + \theta_3)^\star$$
$$\mathcal{E}_n(\mathbb{R}) = \mathcal{L}_n^\star$$
$$\mathcal{E}(\mathbb{R}) = \mathcal{L}^\star$$

Results

- Machine-independent characterizations of classes from Recursive analysis.
- Equivalence between what can be computed by a GPAC and by recursive analysis.
- Can we label $\mathcal{R}ec(\mathbb{R})$ as what is reasonable?

Perspectives

- Understand what complexity means in those models
  - [Ko 91] studies complexity for Recursive Analysis
  - Classes of complexity à la Bellantoni & Cook can be defined as $\mathbb{R}$-recursive functions
  - Complexity in GPAC?
- Extend classical results to $\mathcal{H}^\star$
  - Universal function(s)
  - Fixpoint theorem
  - s_n^m theorem
- Study robustness to perturbations.