Bimachines: an algebraic approach to Turing machine computation

Pedro V. Silva

University of Porto

September 30, 2006

This is part of a joint work with John Rhodes summarized in the preprint *An algebraic analysis of Turing machines and Cook's Theorem leading to a profinite fractal differential equation and a random walk on a deterministic Turing machine.*

The authors thank Jean-Camille Birget for his advice and helpful comments.

lp-mappings

A_1, A_2 - finite nonempty alphabets

A_1^+ - free semigroup on A_1

$\alpha : A_1^+ \to A_2^+$ is an *lp-mapping* if

$$|\alpha(w)| = |w|$$

for every $w \in A_1^+$.

A domain extension

The lp-mapping $\alpha : A_1^+ \to A_2^+$ induces a mapping

$$A_1^* \times A_1 \times A_1^* \quad \to \quad A_2$$

u	a	v

$\alpha \downarrow$

$\ldots$	b	$\ldots$

A domain extension

The lp-mapping $\alpha : A_1^+ \to A_2^+$ induces a mapping

$$A_1^* \times A_1 \times A_1^* \quad \to \quad A_2$$

u	a	v

$$\alpha \downarrow$$

$\dots$	b	$\dots$

$\alpha : A^+ \to A'^+$ is uniquely determined by
$\alpha(\_, \_, \_) : A^* \times A \times A^* \to A'$ and vice-versa.

Right automata

A *right A-automaton* is a triple $\mathcal{A}_R = (I_R, Q_R, S_R)$ where

- Q_R is a set
- $I_R \in Q_R$
- S_R is an A-semigroup acting on Q_R on the right,so

$$(q_R s_R)s'_R = q_R(s_R s'_R).$$

Right automata

A *right A-automaton* is a triple $\mathcal{A}_R = (I_R, Q_R, S_R)$ where

- Q_R is a set
- $I_R \in Q_R$
- S_R is an A-semigroup acting on Q_R on the right,so

$$(q_R s_R)s'_R = q_R(s_R s'_R).$$

The action of S_R on Q_R induces an obvious right action of A^* on Q_R.

Left automata

A *left A-automaton* is a triple $\mathcal{A}_L = (S_L, Q_L, I_L)$ where

- Q_L is a set
- $I_L \in Q_L$
- S_L is an A-semigroup acting on Q_L on the left,so

$$s_L(s'_L q_L) = (s_L s'_L) q_L.$$

Left automata

A *left A-automaton* is a triple $\mathcal{A}_L = (S_L, Q_L, I_L)$ where

- Q_L is a set
- $I_L \in Q_L$
- S_L is an A-semigroup acting on Q_L on the left,so

$$s_L(s'_L q_L) = (s_L s'_L) q_L.$$

The action of S_L on Q_L induces an obvious left action of A^* on Q_L.

Bimachines

An A_1, A_2-*bimachine* is a structure of the form

$$\mathcal{B} = ((I_R, Q_R, S_R), f, (S_L, Q_L, I_L)),$$

where

- (I_R, Q_R, S_R) is a right A-automaton;
- (S_L, Q_L, I_L) is a left A-automaton;
- $f : Q_R \times A \times Q_L \to A'$ a full map (the *output function*).

Bimachines

An A_1, A_2-*bimachine* is a structure of the form

$$\mathcal{B} = ((I_R, Q_R, S_R), f, (S_L, Q_L, I_L)),$$

where

- (I_R, Q_R, S_R) is a right A-automaton;
- (S_L, Q_L, I_L) is a left A-automaton;
- $f : Q_R \times A \times Q_L \to A'$ a full map (the *output function*).

We say that $\mathcal{B}$ is finite if both state sets and semigroups are finite.

From bimachines to lp-mappings

We associate an lp-mapping

$$\alpha_{\mathcal{B}} : A_1^+ \to {A_2}^+$$

to the A_1, A_2-bimachine

$$\mathcal{B} = ((I_R, Q_R, S_R), f, (S_L, Q_L, I_L))$$

by

$$\alpha_{\mathcal{B}}(u, a, v) = f(I_R u, a, vI_L) \quad (u, v \in A^*, a \in A).$$

From bimachines to lp-mappings

We associate an lp-mapping

$$\alpha_{\mathcal{B}} : A_1^+ \to {A_2}^+$$

to the A_1, A_2-bimachine

$$\mathcal{B} = ((I_R, Q_R, S_R), f, (S_L, Q_L, I_L))$$

by

$$\alpha_{\mathcal{B}}(u, a, v) = f(I_R u, a, vI_L) \quad (u, v \in A^*, a \in A).$$

Thus

$$\alpha_{\mathcal{B}}(w) = \prod_{i=1}^{|w|} f(I_R w_i', w_i, w_i'' I_L).$$

From lp-mappings to bimachines

Let $\alpha : A_1^+ \to {A_2}^+$ be an lp-mapping.

Proposition. There exists an A_1, A_2-bimachine $\mathcal{B}_\alpha$ such that:

(i) $\alpha_{\mathcal{B}_\alpha} = \alpha$.

(ii) If $\mathcal{B}'$ is a trim A, A'-bimachine such that $\alpha_{\mathcal{B}'} = \alpha$, then there exists a (surjective) morphism $\varphi : \mathcal{B}' \to \mathcal{B}_\alpha$.

(iii) Up to isomorphism, $\mathcal{B}_\alpha$ is the unique trim A, A'-bimachine satisfying (ii).

From lp-mappings to bimachines

Let $\alpha : A_1^+ \to {A_2}^+$ be an lp-mapping.

Proposition. There exists an A_1, A_2-bimachine $\mathcal{B}_\alpha$ such that:

(i) $\alpha_{\mathcal{B}_\alpha} = \alpha$.

(ii) If $\mathcal{B}'$ is a trim A, A'-bimachine such that $\alpha_{\mathcal{B}'} = \alpha$, then there exists a (surjective) morphism $\varphi : \mathcal{B}' \to \mathcal{B}_\alpha$.

(iii) Up to isomorphism, $\mathcal{B}_\alpha$ is the unique trim A, A'-bimachine satisfying (ii).

We can view $\mathcal{B}_\alpha$ as the *minimum* bimachine of α.

The block product

Let

$$\mathcal{B}^{(i)} = ((I_R^{(i)}, Q_R^{(i)}, S_R^{(i)}), f^{(i)}, (S_L^{(i)}, Q_L^{(i)}, I_L^{(i)}))$$

be an A_i, A_{i+1}-bimachine for $i = 1, 2$. We shall define an A_1, A_3-bimachine

$$\mathcal{B}^{(2)} \Box \mathcal{B}^{(1)} = \mathcal{B}^{(21)} = ((I_R^{(21)}, Q_R^{(21)}, S_R^{(21)}), f^{(21)}, (S_L^{(21)}, Q_L^{(21)}, I_L^{(21)}))$$

called the *block product* of $\mathcal{B}^{(2)}$ and $\mathcal{B}^{(1)}$.

The block product

Let

$$\mathcal{B}^{(i)} = ((I_R^{(i)}, Q_R^{(i)}, S_R^{(i)}), f^{(i)}, (S_L^{(i)}, Q_L^{(i)}, I_L^{(i)}))$$

be an A_i, A_{i+1}-bimachine for $i = 1, 2$. We shall define an A_1, A_3-bimachine

$$\mathcal{B}^{(2)} \Box \mathcal{B}^{(1)} = \mathcal{B}^{(21)} = ((I_R^{(21)}, Q_R^{(21)}, S_R^{(21)}), f^{(21)}, (S_L^{(21)}, Q_L^{(21)}, I_L^{(21)}))$$

called the *block product* of $\mathcal{B}^{(2)}$ and $\mathcal{B}^{(1)}$.

The block product is appropriate do deal with composition:

$$\alpha_{\mathcal{B}^{(2)} \Box \mathcal{B}^{(1)}} = \alpha_{\mathcal{B}^{(2)}} \alpha_{\mathcal{B}^{(1)}}.$$

The semigroups

We define

$$\overline{S_R^{(21)}} = \begin{pmatrix} S_L^{(1)} & 0 \\ {}^{Q_R^{(1)}}S_R^{(2)}{}^{Q_L^{(1)}} & S_R^{(1)} \end{pmatrix}.$$

The semigroups

We define

$$\overline{S_R^{(21)}} = \begin{pmatrix} S_L^{(1)} & 0 \\ {}^{Q_R^{(1)}}S_R^{(2)\,Q_L^{(1)}} & S_R^{(1)} \end{pmatrix}.$$

$\overline{S_R^{(21)}}$ is a semigroup for the product

$$\begin{pmatrix} s_L^{(1)} & 0 \\ g & s_R^{(1)} \end{pmatrix} \begin{pmatrix} s'^{(1)}_L & 0 \\ g' & s'^{(1)}_R \end{pmatrix} = \begin{pmatrix} s_L^{(1)} s'^{(1)}_L & 0 \\ g s'^{(1)}_L + s_R^{(1)} g' & s_R^{(1)} s'^{(1)}_R \end{pmatrix}.$$

The semigroups

We define

$$\overline{S_R^{(21)}} = \begin{pmatrix} S_L^{(1)} & 0 \\ {}^{Q_R^{(1)}}S_R^{(2)}{}^{Q_L^{(1)}} & S_R^{(1)} \end{pmatrix}.$$

$\overline{S_R^{(21)}}$ is a semigroup for the product

$$\begin{pmatrix} s_L^{(1)} & 0 \\ g & s_R^{(1)} \end{pmatrix} \begin{pmatrix} s'^{(1)}_L & 0 \\ g' & s'^{(1)}_R \end{pmatrix} = \begin{pmatrix} s_L^{(1)} s'^{(1)}_L & 0 \\ g s'^{(1)}_L + s_R^{(1)} g' & s_R^{(1)} s'^{(1)}_R \end{pmatrix}.$$

Then we consider the canonical subsemigroup generated by A_1 to get $S_R^{(21)}$.

The states

Let

$$Q_R^{(21)} = {Q_R^{(2)}}^{Q_L^{(1)}} \times Q_R^{(1)}.$$

It will be often convenient to represent the elements of $Q_R^{(21)}$ as 1×2 matrices.

Let

$$I_R^{(21)} = (\gamma_0^{(21)}, I_R^{(1)}),$$

where $\gamma_0^{(21)} \in {Q_R^{(2)}}^{Q_L^{(1)}}$ is defined by $\gamma_0^{(21)}(q_L^{(1)}) = I_R^{(2)}$.

The states

Let

$$Q_R^{(21)} = {Q_R^{(2)}}^{Q_L^{(1)}} \times Q_R^{(1)}.$$

It will be often convenient to represent the elements of $Q_R^{(21)}$ as 1×2 matrices.

Let

$$I_R^{(21)} = (\gamma_0^{(21)}, I_R^{(1)}),$$

where $\gamma_0^{(21)} \in {Q_R^{(2)}}^{Q_L^{(1)}}$ is defined by $\gamma_0^{(21)}(q_L^{(1)}) = I_R^{(2)}$.

The action is given by the natural interpretation of matrix multiplication.

The output function

The output function $f^{(21)} : Q_R^{(21)} \times A_1 \times Q_L^{(21)} \to A_3$ is defined by

$$f^{(21)}(\begin{pmatrix} \gamma & q_R^{(1)} \end{pmatrix}, a, \begin{pmatrix} q_L^{(1)} \\ \delta \end{pmatrix})$$

$$= f^{(2)}(\gamma(aq_L^{(1)}), f^{(1)}(q_R^{(1)}, a, q_L^{(1)}), (q_R^{(1)}a)\delta).$$

The output function

The output function $f^{(21)} : Q_R^{(21)} \times A_1 \times Q_L^{(21)} \to A_3$ is defined by

$$f^{(21)}(\begin{pmatrix} \gamma & q_R^{(1)} \end{pmatrix}, a, \begin{pmatrix} q_L^{(1)} \\ \delta \end{pmatrix})$$

$$= f^{(2)}(\gamma(aq_L^{(1)}), f^{(1)}(q_R^{(1)}, a, q_L^{(1)}), (q_R^{(1)}a)\delta).$$

This completes the definition of the bimachine $\mathcal{B}^{(2)}\Box\mathcal{B}^{(1)}$.

If $\mathcal{B}^{(2)}$ and $\mathcal{B}^{(1)}$ are both finite, so is $\mathcal{B}^{(2)}\Box\mathcal{B}^{(1)}$.

Half associativity

Write

$$\mathcal{B}^{(3(21))} = \mathcal{B}^{(3)} \square (\mathcal{B}^{(2)} \square \mathcal{B}^{(1)}), \quad \mathcal{B}^{((32)1)} = (\mathcal{B}^{(3)} \square \mathcal{B}^{(2)}) \square \mathcal{B}^{(1)}.$$

Lemma. $S_R^{(3(21))} \cong S_R^{((32)1)}$ and $S_L^{(3(21))} \cong S_L^{((32)1)}$.

Half associativity

Write

$$\mathcal{B}^{(3(21))} = \mathcal{B}^{(3)}\Box(\mathcal{B}^{(2)}\Box\mathcal{B}^{(1)}), \quad \mathcal{B}^{((32)1)} = (\mathcal{B}^{(3)}\Box\mathcal{B}^{(2)})\Box\mathcal{B}^{(1)}.$$

Lemma. $S_R^{(3(21))} \cong S_R^{((32)1)}$ and $S_L^{(3(21))} \cong S_L^{((32)1)}$.

Theorem. $(\mathcal{B}^{(3)}\Box\mathcal{B}^{(2)})\Box\mathcal{B}^{(1)}$ is a quotient of $\mathcal{B}^{(3)}\Box(\mathcal{B}^{(2)}\Box\mathcal{B}^{(1)})$.

Iteration of the block product

We shall choose bracketing from left to right, that is, priority is assumed to hold from left to right:

$$\mathcal{B}^{[n,1]} = ((\ldots(\mathcal{B}^{(n)} \Box \mathcal{B}^{(n-1)}) \Box \mathcal{B}^{(n-2)}) \Box \ldots) \Box \mathcal{B}^{(1)}.$$

Iteration of the block product

We shall choose bracketing from left to right, that is, priority is assumed to hold from left to right:

$$\mathcal{B}^{[n,1]} = ((\ldots(\mathcal{B}^{(n)}\Box\mathcal{B}^{(n-1)})\Box\mathcal{B}^{(n-2)})\Box\ldots)\Box\mathcal{B}^{(1)}.$$

The state sets can be naturally written as direct products:

$$Q_R^{[n,1]} = \quad {Q_R^{(n)}}^{Q_L^{(1)}\times Q_L^{(2)}\times\ldots\times Q_L^{(n-1)}} \times {Q_R^{(n-1)}}^{Q_L^{(1)}\times Q_L^{(2)}\times\ldots\times Q_L^{(n-2)}}$$

$$\times\ldots\times {Q_R^{(2)}}^{Q_L^{(1)}} \times Q_R^{(1)}.$$

A sequential action

Theorem. The right action of A_1^+ on $Q_R^{[n,1]}$ is sequential.

This means that if

$$u(\gamma_n, \ldots, \gamma_1) = (\gamma'_n, \ldots, \gamma'_1),$$

then γ'_j depends on $\gamma_j, \ldots, \gamma_1$ and u only.

A sequential action

Theorem. The right action of A_1^+ on $Q_R^{[n,1]}$ is sequential.

This means that if

$$u(\gamma_n, \ldots, \gamma_1) = (\gamma'_n, \ldots, \gamma'_1),$$

then γ'_j depends on $\gamma_j, \ldots, \gamma_1$ and u only.

This leads naturally to a tree representation of the action and allows the definition of profinite limits.

Our model (informal)

We are interested in deterministic Turing machines that halt for all inputs, particularly those that can solve NP-complete problems. In comparison with the most standard models, our model presents three particular features:

Our model (informal)

We are interested in deterministic Turing machines that halt for all inputs, particularly those that can solve NP-complete problems. In comparison with the most standard models, our model presents three particular features:

- the "tape" is potentially infinite in *both* directions and has a distinguished cell named *the origin*;

Our model (informal)

We are interested in deterministic Turing machines that halt for all inputs, particularly those that can solve NP-complete problems. In comparison with the most standard models, our model presents three particular features:

- the "tape" is potentially infinite in *both* directions and has a distinguished cell named *the origin*;
- the origin contains the symbol $\#$ until the very last move of the computation, and $\#$ appears in no other cell;

Our model (informal)

We are interested in deterministic Turing machines that halt for all inputs, particularly those that can solve NP-complete problems. In comparison with the most standard models, our model presents three particular features:

- the "tape" is potentially infinite in *both* directions and has a distinguished cell named *the origin*;
- the origin contains the symbol $\#$ until the very last move of the computation, and $\#$ appears in no other cell;
- the machine always halts in one of a very restricted set of configurations: $B^*Y^+B^*$ (yes) or $B^*N^+B^*$ (no).

Word formalism

We replace the classical model of "tape" and "control head" by a formalism based on words. Let

$$A' = A \cup \{a^q \mid a \in A, q \in Q\}$$

be the *extended tape alphabet*.

The exponent q on a symbol acnowledges the present scanning of the corresponding cell by the control head, under state q.

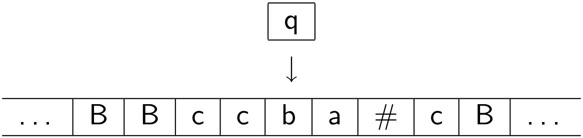
Word formalism

We replace the classical model of "tape" and "control head" by a formalism based on words. Let

$$A' = A \cup \{a^q \mid a \in A, q \in Q\}$$

be the *extended tape alphabet*.

The exponent q on a symbol acnowledges the present scanning of the corresponding cell by the control head, under state q.



$$BBccb^q a\#cB$$

The one-move mapping

ID - instantaneous descriptions (factors of a tape word)

The Turing machine $\mathcal{T}$ induces a mapping $\beta : ID \to ID$ as follows:
Let $w \in ID$. In the absence of states, let $\beta(w) = w$.

The one-move mapping

ID - instantaneous descriptions (factors of a tape word)

The Turing machine $\mathcal{T}$ induces a mapping $\beta : ID \to ID$ as follows:
Let $w \in ID$. In the absence of states, let $\beta(w) = w$.
Suppose now that $w = ua^q v$ with $a \in A$ and $q \in Q$.

- if $\delta(q, a) = b \in \{Y, N\}$, let $\beta(w) = ubv$;
- if $\delta(q, a) = (p, b, R)$ and c is the first letter of $v = cv'$, let $\beta(w) = ubc^p v'$;
- if $\delta(q, a) = (p, b, R)$ and $v = 1$, let $\beta(w) = ubB^p$;
- if $\delta(q, a) = (p, b, L)$ and c is the last letter of $u = u'c$, let $\beta(w) = u'c^p bv$;
- if $\delta(q, a) = (p, b, R)$ and $u = 1$, let $\beta(w) = B^p bv$.

The one-move lp-mapping

$\beta_0 : ID \to ID$ is defined by

$$\beta_0(w) = \begin{cases} \beta(w) & \text{if } |\beta(w)| = |w| \\ w' & \text{if } |\beta(w)| = |w| + 1 \text{ and } \beta(w) = w'B^p; \\ w' & \text{if } |\beta(w)| = |w| + 1 \text{ and } \beta(w) = B^p w'. \end{cases}$$

The one-move lp-mapping

$\beta_0 : ID \to ID$ is defined by

$$\beta_0(w) = \begin{cases} \beta(w) & \text{if } |\beta(w)| = |w| \\ w' & \text{if } |\beta(w)| = |w| + 1 \text{ and } \beta(w) = w'B^p; \\ w' & \text{if } |\beta(w)| = |w| + 1 \text{ and } \beta(w) = B^p w'. \end{cases}$$

Alternatively, we can say that $\beta_0(w)$ is obtained from $\beta(BwB)$ by removing the first and the last letter.

The one-move lp-mapping

$\beta_0 : ID \to ID$ is defined by

$$\beta_0(w) = \begin{cases} \beta(w) & \text{if } |\beta(w)| = |w| \\ w' & \text{if } |\beta(w)| = |w| + 1 \text{ and } \beta(w) = w'B^p; \\ w' & \text{if } |\beta(w)| = |w| + 1 \text{ and } \beta(w) = B^p w'. \end{cases}$$

Alternatively, we can say that $\beta_0(w)$ is obtained from $\beta(BwB)$ by removing the first and the last letter.

We can deduce $\beta(w)$ from $\beta_0(BwB)$ and, more generally, $\beta^n(w)$ from $\beta_0(B^n w B^n)$.

The one-move bimachine

The A', A'-bimachine

$$\mathcal{B}_{\mathcal{T}} = ((I_R, Q_R, S_R), f, (S_L, Q_L, I_L))$$

is defined as follows:

- $Q_R = A' \cup \{I_R\}$, $Q_L = A' \cup \{I_L\}$;
- $S_R = A'$ is a right zero semigroup ($ab = b$);
- $S_L = A'$ is a left zero semigroup ($ab = a$);

The one-move bimachine

The A', A'-bimachine

$$\mathcal{B}_{\mathcal{T}} = ((I_R, Q_R, S_R), f, (S_L, Q_L, I_L))$$

is defined as follows:

- $Q_R = A' \cup \{I_R\}$, $Q_L = A' \cup \{I_L\}$;
- $S_R = A'$ is a right zero semigroup ($ab = b$);
- $S_L = A'$ is a left zero semigroup ($ab = a$);
- the action $Q_R \times S_R \to Q_R$ is defined by $q_R a = a$;
- the action $S_L \times Q_L \to Q_L$ is defined by $aq_L = a$

The one-move bimachine

For the output function: given $q_R \in Q_R$, $a \in A'$ and $q_L \in Q_L$, let

$$f(q_R, a, q_L) = \beta_0(q_R, a, q_L)$$

(replacing I_R or I_L by B if necessary).

The one-move bimachine

For the output function: given $q_R \in Q_R$, $a \in A'$ and $q_L \in Q_L$, let

$$f(q_R, a, q_L) = \beta_0(q_R, a, q_L)$$

(replacing I_R or I_L by B if necessary). If $q_R a q_L \in ID$, then $q_R a q_L$ will encode the situation of three consecutive tape cells at a certain moment. Then $f(q_R, a, q_L)$ describes the situation of the middle cell after one move of $\mathcal{T}$.

The one-move bimachine

For the output function: given $q_R \in Q_R$, $a \in A'$ and $q_L \in Q_L$, let

$$f(q_R, a, q_L) = \beta_0(q_R, a, q_L)$$

(replacing I_R or I_L by B if necessary). If $q_R a q_L \in ID$, then $q_R a q_L$ will encode the situation of three consecutive tape cells at a certain moment. Then $f(q_R, a, q_L)$ describes the situation of the middle cell after one move of $\mathcal{T}$.

Proposition. Let $\mathcal{T}$ be a TM with one-move lp-mapping β_0. Then $\alpha_{\mathcal{B}_{\mathcal{T}}}(w) = \beta_0(w)$ for every $w \in ID$.

The projective limit

We assume now that $\mathcal{B}^{(1)} = \mathcal{B}^{(2)} = \mathcal{B}^{(3)} = \ldots$ are countably many copies of the one-move lp-mapping of a TM.

Lemma. $\{(I_R^{[n,1]}, Q_R^{[n,1]}, S_R^{[n,1]}) \mid n \geq 1\}$ with the natural morphisms constitute a projective system of right A', A'-automata.

The projective limit

We assume now that $\mathcal{B}^{(1)} = \mathcal{B}^{(2)} = \mathcal{B}^{(3)} = \ldots$ are countably many copies of the one-move lp-mapping of a TM.

Lemma. $\{(I_R^{[n,1]}, Q_R^{[n,1]}, S_R^{[n,1]}) \mid n \geq 1\}$ with the natural morphisms constitute a projective system of right A', A'-automata.

$(I_R^{\omega}, Q_R^{\omega}, S_R^{\omega})$ is its projective limit.

There are dual results for the left automata. We write

$$\mathcal{B}^{\omega} = ((I_R^{\omega}, Q_R^{\omega}, S_R^{\omega}), f^{\omega}, (S_L^{\omega}, Q_L^{\omega}, I_L^{\omega})).$$

The differential equation

Given $u, v \in {A'}^+$ and $a \in A'$, we define

$$f^{\omega}(I_R^{\omega}u, a, vI_L^{\omega}) = \lim_{n \to \infty} f^{[n,1]}(I_R^{[n,1]}B^n u, a, vB^n I_L^{[n,1]}).$$

The differential equation

Given $u, v \in {A'}^+$ and $a \in A'$, we define

$$f^\omega(I_R^\omega u, a, vI_L^\omega) = \lim_{n \to \infty} f^{[n,1]}(I_R^{[n,1]} B^n u, a, vB^n I_L^{[n,1]}).$$

$\mathcal{B}^\omega$ satisfies the following property (the differential equation):

Theorem. $\mathcal{B}^\omega \cong \mathcal{B}^\omega \Box \mathcal{B}$.

Thus $\mathcal{B}^\omega$ is a *self-similar* bimachine.

The project

The profinite bimachine $\mathcal{B}^\omega$ encodes the full computational power of $\mathcal{T}$, with appropriate equivalents of time and space functions.

Its self-similarity opens new perspectives to the study of complexity, allowing the use of recursion at an algebraic level (see Grigorchuk et al. on self-similar groups).

The project

The profinite bimachine $\mathcal{B}^\omega$ encodes the full computational power of $\mathcal{T}$, with appropriate equivalents of time and space functions.

Its self-similarity opens new perspectives to the study of complexity, allowing the use of recursion at an algebraic level (see Grigorchuk et al. on self-similar groups).

We hope that this entirely new approach to Turing machine computation may eventually lead to new results in complexity theory, possibly a proof for P $\neq$ NP.

A workshop devoted to this approach was held in June at the University of Berkeley.