Enzymatic numerical P system using elementary arithmetic operations

school

Name Alberto Le

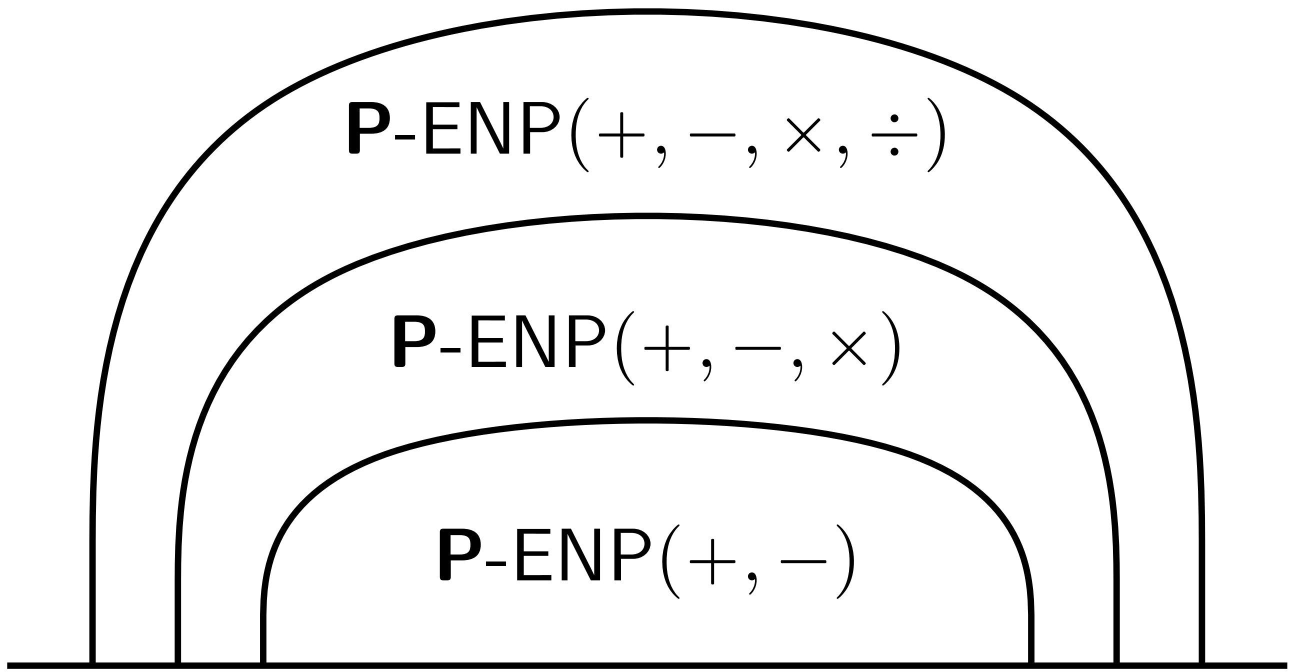
speaker Antonio E. Porreca, Claudio Zandron

School Università degli Studi di Milano-Bicocca

Subject 14th Conference on Membrane Computing

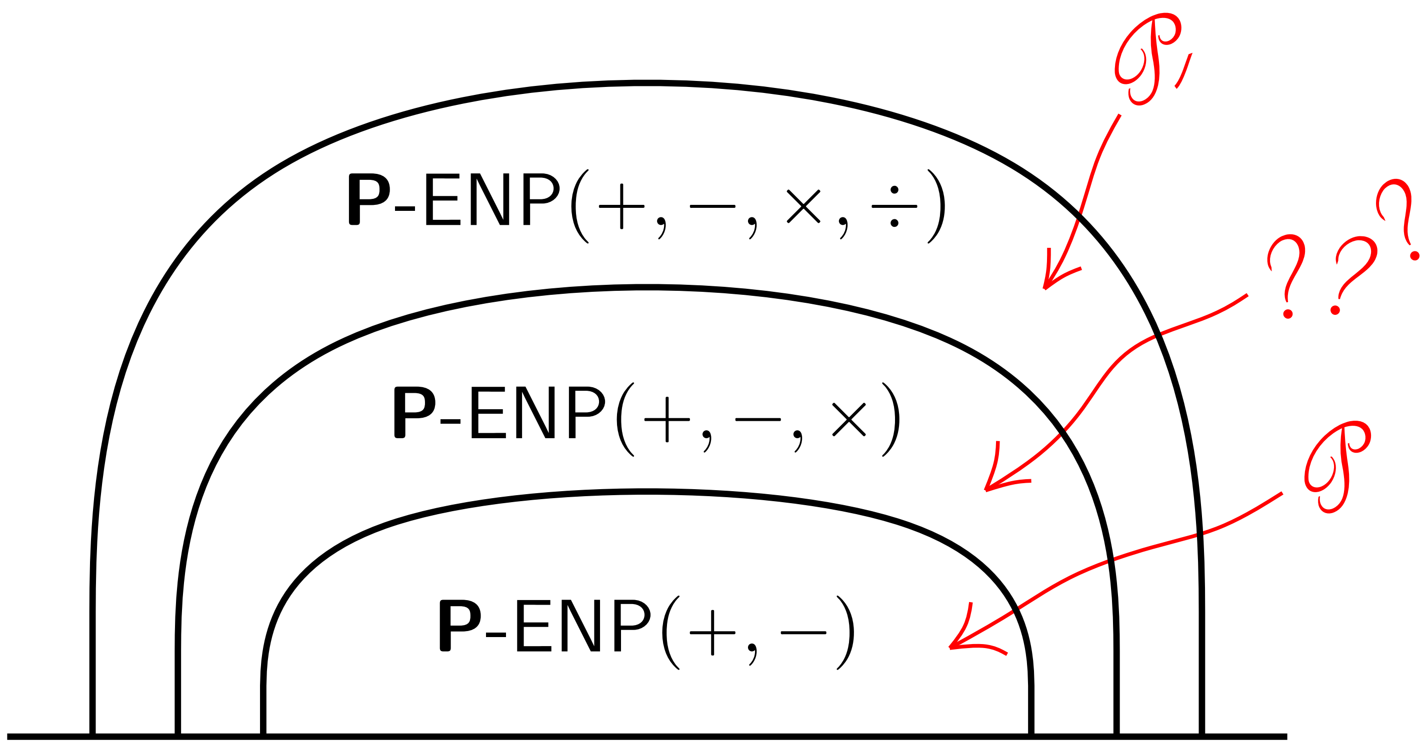
Summary

- We study the computational complexity of enzymatic numerical P systems as recognisers
- We show that the power of polynomial-time ENPs changes depending on which operations are allowed on the LHS of rules



Summary

- We study the computational complexity of enzymatic numerical P systems as recognisers
- We show that the power of polynomial-time ENPs changes depending on which operations are allowed on the LHS of rules



Numerical P sy

- Tree-like membrane structure
- Variables with non-negative integer values
- Programs of the form

$$F(x_1, \ldots, x_m) \rightarrow a_1|y_1 + \cdots + a_n|y_m$$

Numerical P sy

- Tree-like membrane structure
- Variables with non-negative integer values
- Programs of the form

$$F(x_1, \ldots, x_m) \to a_1|y_1 + \cdots + a_n|y_m$$

production function

- Tree-like membrane structure
- Variables with non-negative integer values
- Programs of the form

$$F(x_1, \ldots, x_m) \rightarrow a_1|y_1 + \cdots + a_n|y_m$$

re
protocol

- Tree-like membrane structure
- Variables with non-negative integer values
- Programs of the form

$$F(x_1, \ldots, x_m) \to a_1|y_1 + \cdots + a_n|y_m$$

- Variables on the LHS are zeroed
- Variable y_j on the RHS gets $a_j/\Sigma_i a_i$ of the result

- Further type of program:

$$F(x_1, \ldots, x_m)|_e \rightarrow a_1|y_1 + \cdots + a_n|y_m$$

- Further type of program:

$$F(x_1, \ldots, x_m)|_e \longrightarrow a_1|y_1 + \cdots + a_n|y_m$$

enzyme

- Further type of program:

$$F(x_1, \ldots, x_m)|_e \rightarrow a_1|y_1 + \cdots + a_n|y_m$$

- Variable e must not occur in PF or RHS

- Further type of program:

$$F(x_1, \ldots, x_m)|_e \to a_1|y_1 + \cdots + a_n|y_m$$

- Variable e must not occur in PF or RHS
- The program is enabled iff $e > \min\{x_1, \ldots, x_m\}$

Parallelism policie

- Sequential: one program per region
- All-parallel: each enabled program is executed
- One-parallel: each variable may be used only once
- In all-parallel and one-parallel mode
  the ENPs can always be flattened

Parallelism policie

- Sequential: one program per region
- All-parallel: each enabled program is executed
- One-parallel: each variable may be used only once
- In all-parallel and one-parallel mode
  the ENPs can always be flattened

here we assume this and this!

Recogniser ENP sy

- Let $L \subseteq \{0,1\}^\star$
- Let Π be an ENP with variables *accept* and *reject*
- For $x \in L$, initialise Π with $1x$ in an input variable
- Assume Π reaches a stable configuration
- If $x \in L$, then $accept = 1$ and $reject = 0$
- If $x \notin L$, then $accept = 0$ and $reject = 1$

- Let $L \subseteq \{0,1\}^\star$
- Let Π be an ENP with variables *accept* and *reject*
- For $x \in L$, initialise Π with $1x$ in an input variable
- Assume Π reaches a stable configuration
- If $x \in L$, then $accept = 1$ and $reject = 0$
- If $x \notin L$, then $accept = 0$ and $reject = 1$

the P sy

$\mathbf{P}\text{-ENP}(+, -)$

$\mathbf{P}\text{-ENP}(+, -, \times)$

$\mathbf{P}\text{-ENP}(+, -, \times, \div)$

Complexity classe

$\mathbf{P}$-ENP$(+,-)$

$\mathbf{P}$-ENP$(+,-,\times)$

$\mathbf{P}$-ENP$(+,-,\times,\div)$

polynomial time

P-ENP$(+,-)$

P-ENP$(+,-,\times)$

P-ENP$(+,-,\times,\div)$

o

allowed in the PFs

$\mathbf{P}\text{-ENP}(+, -)$

$\mathbf{P}\text{-ENP}(+, -, \times)$

$\mathbf{P}\text{-ENP}(+, -, \times, \div)$

non-negative subtraction!

Random acce

Infinitely many registers $(r_i : i \in \mathbf{N})$

- $\ell\colon\ r_i \coloneqq k$
- $\ell\colon\ r_i \coloneqq r_j$
- $\ell\colon\ r_i \coloneqq r_{r_j}$
- $\ell\colon\ r_i \coloneqq r_j \bullet r_k$
- ℓ: if $r_i \neq 0$ then ℓ_1 else ℓ_2
- ℓ: accept
- ℓ: reject

Infinitely many registers $(r_i : i \in \mathbf{N})$

- ℓ: $r_i := k$
- ℓ: $r_i := r_j$
- ℓ: $r_i := r_{r_j}$
- ℓ: $r_i := r_j \bullet r_k$
- ℓ: if $r_i \neq 0$ then ℓ_1 else ℓ_2
- ℓ: accept
- ℓ: reject

can be $+ - \times \div$

has $O(1)$ co

$\mathbf{P}$-RAM$(+, -)$

$\mathbf{P}$-RAM$(+, -, \times, \div)$

$$\mathbf{P}\text{-RAM}(+,-) = \mathbf{P}$$

$$\mathbf{P}\text{-RAM}(+,-,\times,\div)$$

$$\mathbf{P}\text{-RAM}(+,-) = \mathbf{P}$$

$$\mathbf{P}\text{-RAM}(+,-,\times,\div) = \mathbf{PSPACE}$$

Avoiding indirect addre

- Indirect addressing (and unbounded number of registers) can be avoided in RAMs
- Represent the registers of a machine M as a single base-b number
- Here $b = 1 +$ largest number stored by M

$$r = b^{m-1} r_{m-1} + b^{m-2} r_{m-2} + \cdots + b^1 m_1 + b^0 r_0$$

Avoiding indirect addre

- Indirect addressing (and unbounded number of registers) can be avoided in RAMs
- Represent the registers of a machine M as a single base-b number
- Here $b = 1 +$ largest number stored by M

$$r = b^{m-1} r_{m-1} + b^{m-2} r_{m-2} + \cdots + b^1 m_1 + b^0 r_0$$

register m-1

Avoiding indirect addre

- Indirect addressing (and unbounded number of registers) can be avoided in RAMs
- Represent the registers of a machine M as a single base-b number
- Here $b = 1 +$ largest number stored by M

$$r = b^{m-1} r_{m-1} + b^{m-2} r_{m-2} + \cdots + b^1 m_1 + b^0 r_0$$

register 1

Proposition. If M is a RAM$(+,-)$ working in t steps on input $x \in \mathbf{N}$, then $b = 2^t x + 1$ obtained by repeated doubling

Proposition. If M is a RAM$(+,-,\times,\div)$ working in t steps on input $x \in \mathbf{N}$, then $b = x^{2^t} + 1$ obtained by repeated squaring

Proposition. The operations $x \times y$ and $x \div y$ can be executed in time $O(|x|^2)$ and $O(|y|^2)$ by a RAM$(+,-)$ by repeated doubling

Proposition. The operation x^y can be executed in polynomial time wrt $O(|y|^2)$ by a RAM$(+,-,\times,\div)$ by repeated squaring

Proposition. The operation x^y can be executed in time $O(y^2|y|^2|x|^2)$ by a RAM$(+,-)$ by repeated squaring

Example: implementing ex

$e := y$
$z := 1$
while $e > 0$ **do**
    $\{x^e \times z = x^y\}$
    $p := 1$
    $p' := 2$
    $a := x$
    $a' := x \times x$
    while $p' \leqslant e$ **do**
        $p := p'$
        $p' := p' + p'$
        $a := a'$
        $a' := a' \times a'$
    end
    $\{e - p \leqslant e/2\}$
    $e := e - p$
    $z := z \times a$
end

$|y|$ time

$|y|$ time

Assignment of a constant "$r_i := c$"

$$z := (r \div b^i) \mod b$$
$$r := r - (z \times b^i) + (c \times b^i)$$

Copying the value of a register "$r_i := r_j$"

$$y := (r \div b^j) \mod b$$
$$z := (r \div b^i) \mod b$$
$$r := r - (z \times b^i) + (y \times b^i)$$

Copying the value of a register w/i.a. "$r_i := r_{r_j}$"

$$y := (r \div b^j) \mod b$$
$$y' := (r \div b^y) \mod b$$
$$z := (r \div b^i) \mod b$$
$$r := r - (z \times b^i) + (y' \times b^i)$$

Arithmetical operations "$r_i := r_j \bullet r_k$"

$$y_1 := (r \div b^j) \mod b$$
$$y_2 := (r \div b^k) \mod b$$
$$y := y_1 \bullet y_2$$
$$z := (r \div b^i) \mod b$$
$$r := r - (z \times b^i) + (y \times b^i)$$

Conditional jump "if $r_i \neq 0$ then ℓ_1 else ℓ_2"

$$y := (r \div b^i) \mod b$$
if $y \neq 0$ then ℓ'_1 else ℓ'_2

Theorem 1. Each RAM without indirect addressing can be simulated by an EN P system using the same number of steps

Theorem 1. Each RAM without indirect addressing can be simulated by an EN P system using the same number of steps

Assignment of a constant "$r_i := c$"

$$0r_i + k + z|_{p_\ell} \to 1|r_i$$

$$p_\ell \to 1|p_{\ell+1}$$

Theorem 1. Each RAM without indirect addressing can be simulated by an EN P system using the same number of steps

Assignment of a constant "$r_i := c$"

$$0r_i + k + z|_{p_\ell} \to 1|r_i$$
$$p_\ell \to 1|p_{\ell+1}$$

Copying the value of a register "$r_i := r_j$"

$$0r_i + 2r_j + z|_{p_\ell} \to 1|r_i + 1|r_j$$
$$p_\ell \to 1|p_{\ell+1}$$

Arithmetical operations "$r_i \coloneqq r_j \bullet r_k$"

$$0 r_i + r_j \bullet r_k + z|_{p_\ell} \to 1|r_i$$

$$r_j + z|_{p_\ell} \to 1|r_j$$

$$r_k + z|_{p_\ell} \to 1|r_k$$

$$p_\ell \to 1|p_{\ell+1}$$

Arithmetical operations "$r_i \coloneqq r_j \bullet r_k$"

$$0 r_i + r_j \bullet r_k + z|_{p_\ell} \to 1|r_i$$
$$r_j + z|_{p_\ell} \to 1|r_j$$
$$r_k + z|_{p_\ell} \to 1|r_k$$
$$p_\ell \to 1|p_{\ell+1}$$

Conditional jump "if $r_i \neq 0$ then ℓ_1 else ℓ_2"

$$p_\ell \to 1|p_{\ell_1}$$
$$r_i - 1|_{p_\ell} \to 1|p_{\ell_1}$$
$$r_i + 1|_{p_\ell} \to 1|p_{\ell_2}$$

QED

repeat
  save the current values of the variables
  compute the variations due to p_1 *(if applicable)*
  ⋮
  compute the variations due to p_h *(if applicable)*
  compute the new values of the variables
until *a final configuration is reached*
if Π *accepted* **then**
  accept
else
  reject
end

Example: $f(x_{i_1}, \ldots, x_{i_k})|_e \to a_1|x_1 + \cdots + a_m|x_m$

Example: $f(x_{i_1},\ldots,x_{i_k})|_e \to a_1|x_1 + \cdots + a_m|x_m$

if $e > x_{i_1}$ **or** $e > x_{i_1}$ **or** $\cdots$ **or** $e > x_{i_k}$ **then**

$f := f(x_{i_1},\ldots,x_{i_k})$

$x'_{i_1} := 0$

$\vdots$

$x'_{i_k} := 0$

$u := f \div (a_1 + \cdots + a_m)$

$\Delta_1 := \Delta_1 + a_1 u$

$\vdots$

$\Delta_m := \Delta_m + a_m u$

end

Theorem 2. An ENP$(+,-)$ can be simulated in polynomial time by a RAM$(+,-)$, and an ENP$(+,-,\times,\div)$ by a RAM$(+,-,\times,\div)$

Theorem 2. An ENP$(+,-)$ can be simulated in polynomial time by a RAM$(+,-)$, and an ENP$(+,-,\times,\div)$ by a RAM$(+,-,\times,\div)$

Theorem 3.

$$\mathbf{P}\text{-ENP}(+,-) = \mathbf{P}\text{-RAM}(+,-) = \mathbf{P}$$
$$\mathbf{P}\text{-ENP}(+,-,\times,\div) = \mathbf{P}\text{-RAM}(+,-,\times,\div) = \mathbf{PSPACE}$$

- $\mathbf{P}$-ENP$(+, -, \times) = ???$
- ($\mathbf{P}$-RAM$(+, -, \times)$ is also unknown)
- What about sequential mode?
- What about one-parallem mode?
- What about ENPs without enzymes?

Thanks for your attention!

Vă mulțumim pentru atenție!

Спасибо за внимание!

Any que