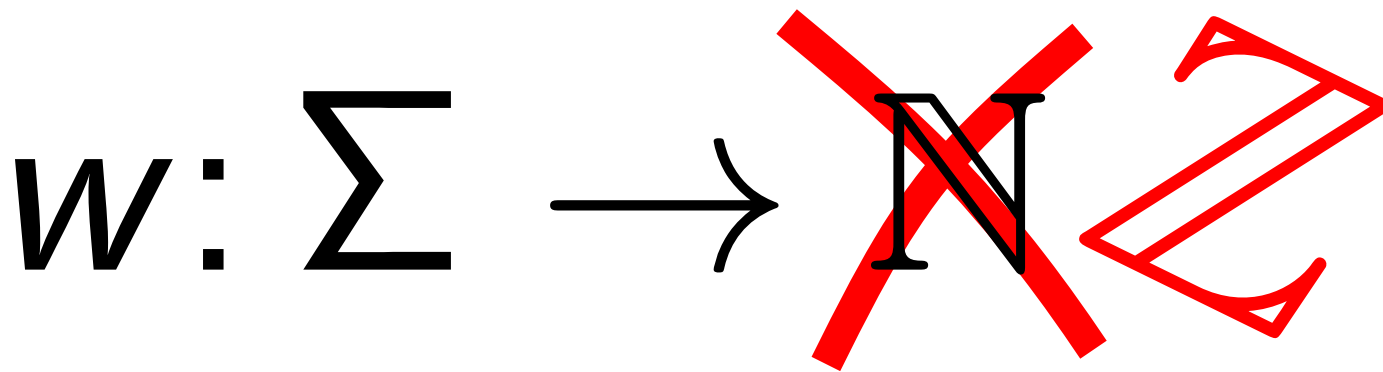


P systems with hybrid sets



Omar Belingheri, Antonio E. Porreca, Claudio Zandron
Università degli Studi di Milano-Bicocca, Italy

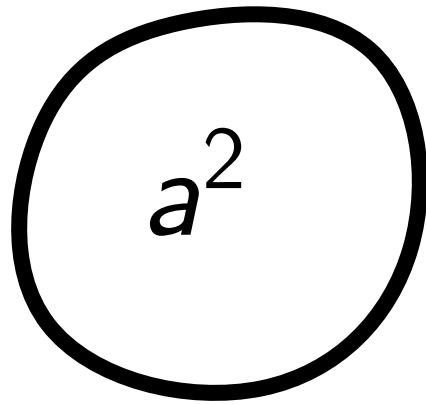
Inspiration & other works

Gheorghe Păun, **Some quick research topics**,
*Proceedings of the Thirteenth Brainstorming
Week on Membrane Computing*

Rudi Freund, Sergiu Ivanov, Sergey Verlan,
**P systems with generalized multisets over totally
ordered abelian groups**, *Proceedings of the 16th
International Conference on Membrane
Computing (CMC16)*

“Negative objects”

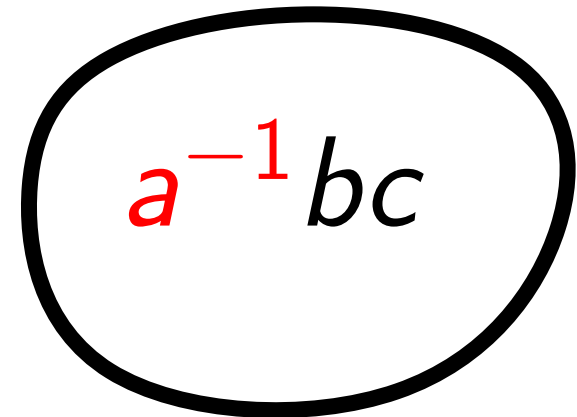
$$a^3 \rightarrow bc$$



A circle containing the expression a^2 .

application
of the rule

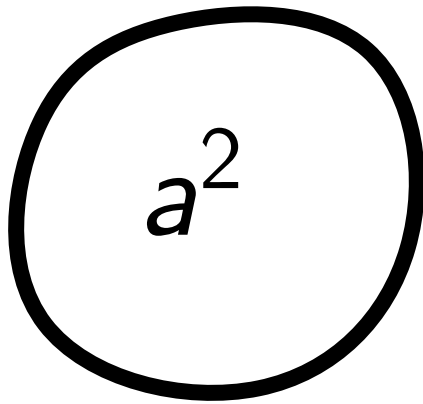




A circle containing the expression $a^{-1}bc$, where the a^{-1} is in red.


¿When do we stop?

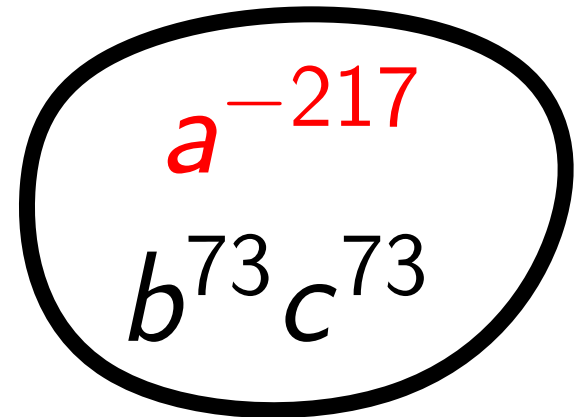
$$a^3 \rightarrow bc$$



A circle containing the expression a^2 .

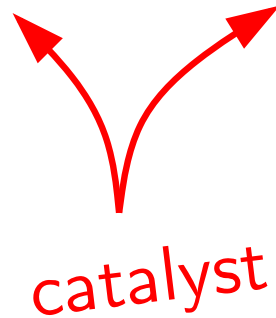
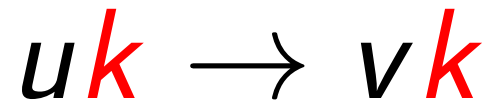
73 applications
of the rule





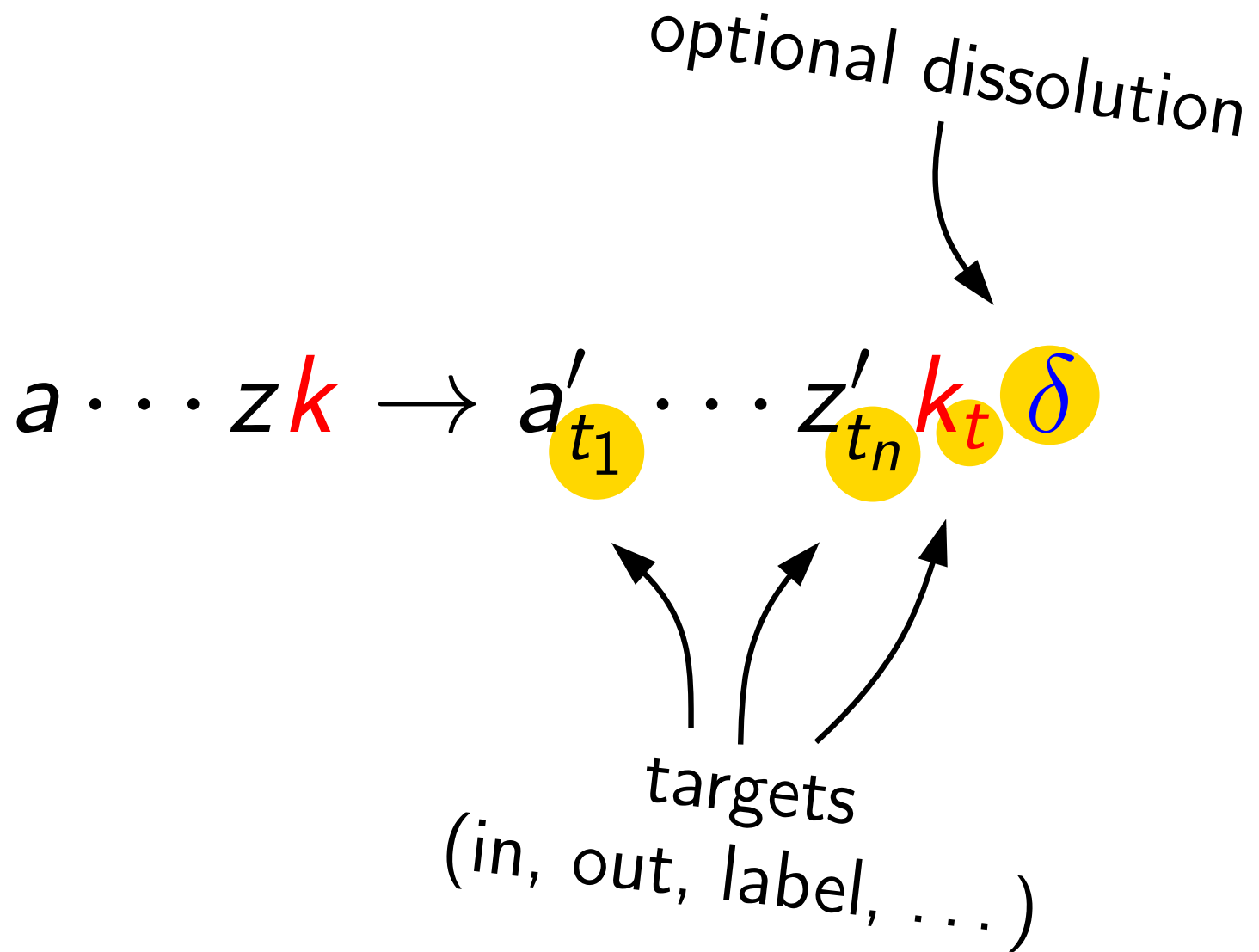
An oval containing the expression $a^{-217} b^{73} c^{73}$. The a^{-217} term is written in red.

Proposal: have catalyst objects



Catalysts obey mass conservation
and cannot have negative multiplicity

General rule form



Consequence: rules become context-free

$$uk \rightarrow vk$$



$$k \rightarrow vu^{-1}k$$

Consequence: no zero test

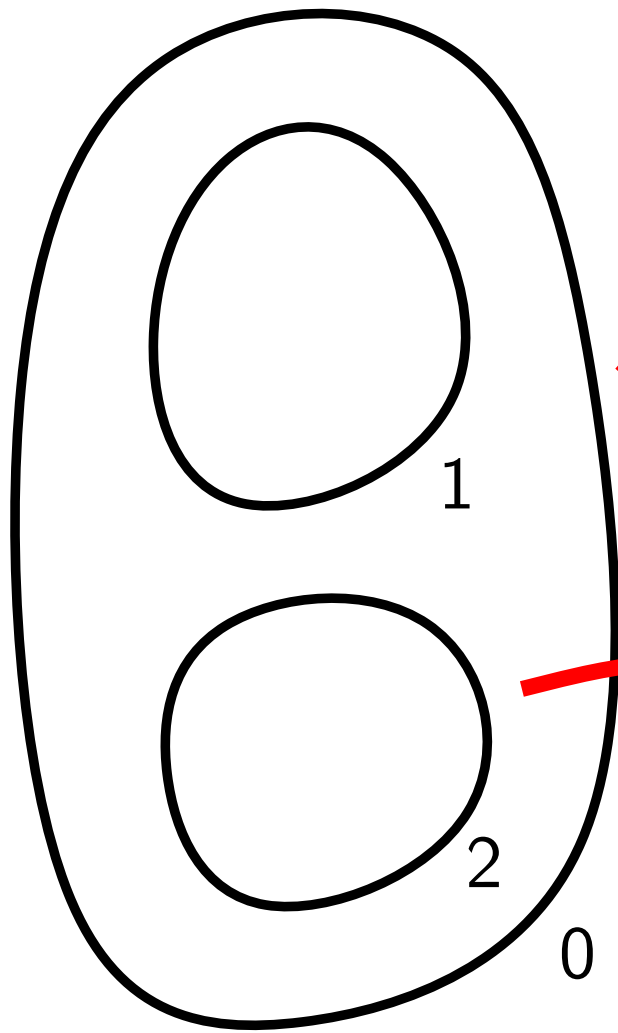
$l_1 : \text{add}(r), l_2, l_3$

$l_1 : \text{sub}(r), l_2, \text{abort}$

$l_1 : \text{halt}$

A subset of the registers must be null at the end of legitimate computations

Simulation by partially blind machines



“positive” a 's in region 0

“negative” a 's in region 0

“positive” b 's in region 0

“negative” b 's in region 0

⋮

“negative” z 's in region 2

registers for catalysts

output registers

Simulation algorithm

*just look at
the catalysts*



while not halted **do**

nondeterministically choose a multiset of rules to apply

for each chosen rule $uk \rightarrow vk$ **do**

add u to the corresponding “negative” registers

add v to the corresponding “positive” registers

for each output symbol a **do**

nondeterministically guess if $\#a$ is negative

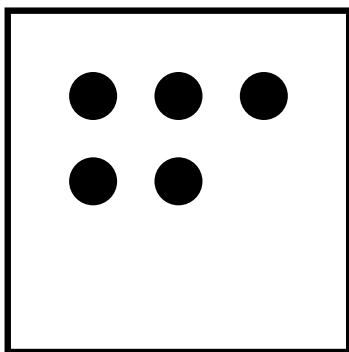
if guessed negative **then**

compute Δa in the negative output register

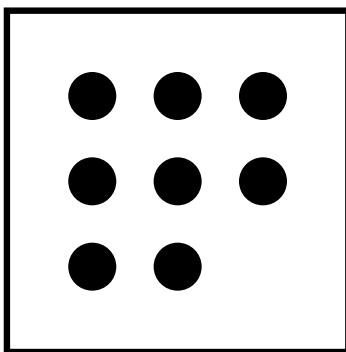
else

compute Δa in the positive output register

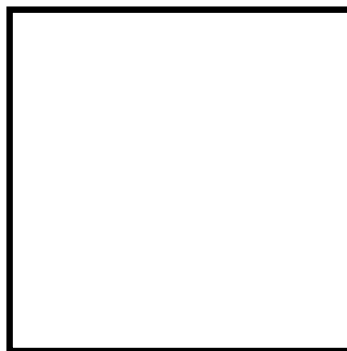
Computing Δa



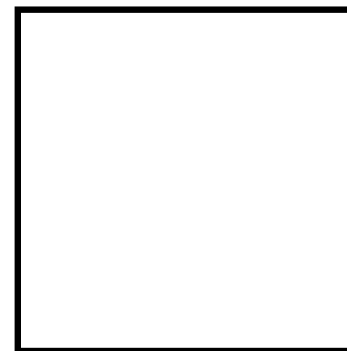
“positive” a 's
in region h



“negative” a 's
in region h

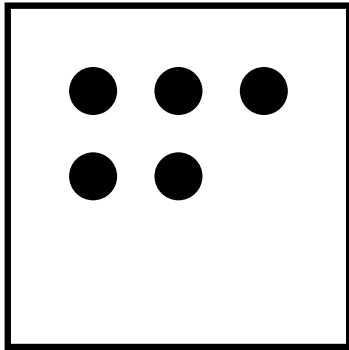


“positive”
output a 's

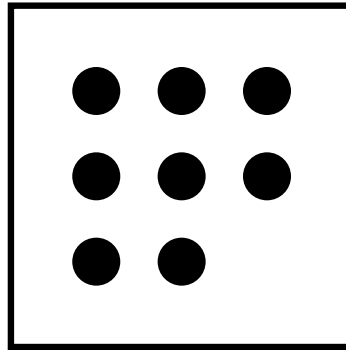


“negative”
output a 's

Computing Δa

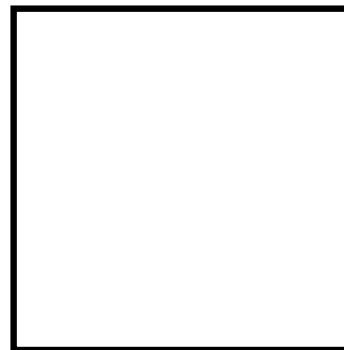


“positive” a 's
in region h

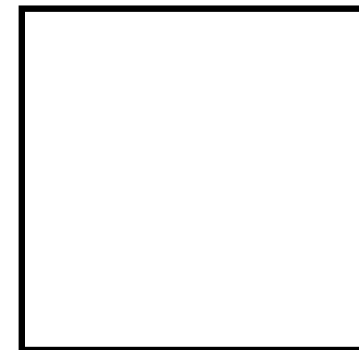


“negative” a 's
in region h

*nondeterministic
guess:
 $\#a < 0$*

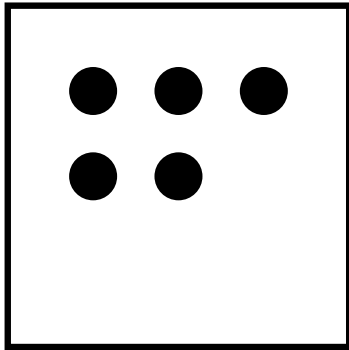


“positive”
output a 's

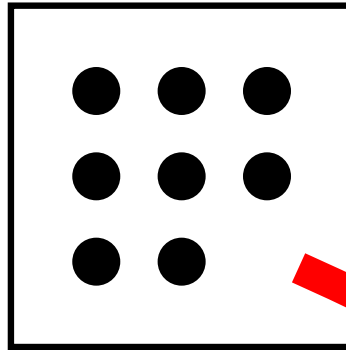


“negative”
output a 's

Computing Δa



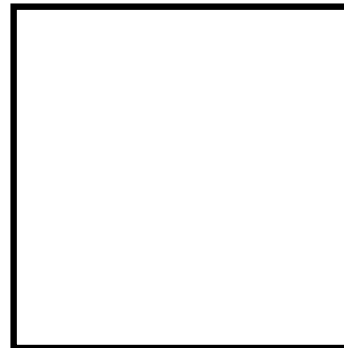
“positive” a 's
in region h



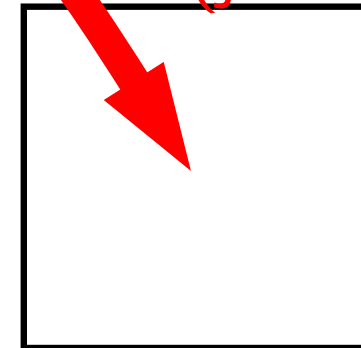
“negative” a 's
in region h

*nondeterministic
guess:
 $\#a < 0$*

*copy until empty
(just guess)*

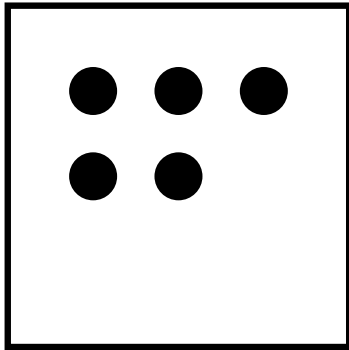


“positive”
output a 's

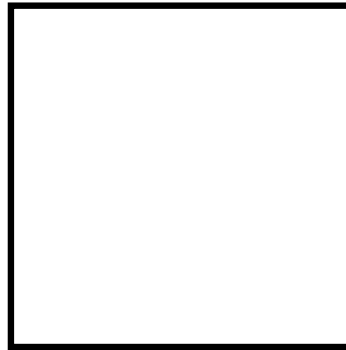


“negative”
output a 's

Computing Δa

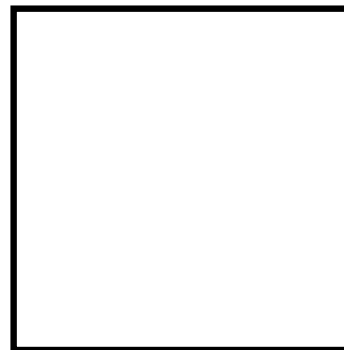


“positive” a 's
in region h

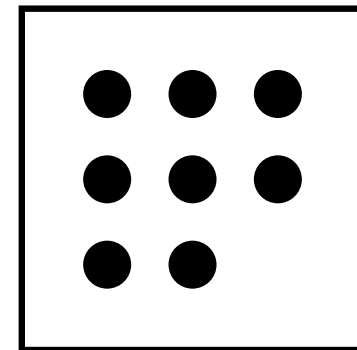


“negative” a 's
in region h

*nondeterministic
guess:
 $\#a < 0$*

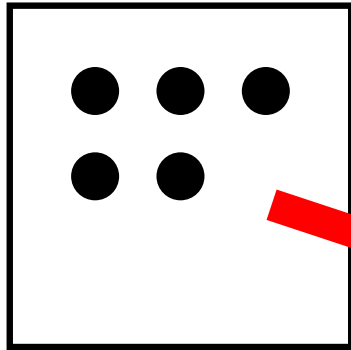


“positive”
output a 's

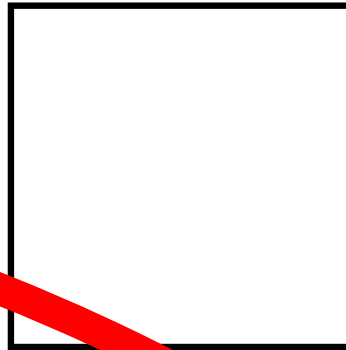


“negative”
output a 's

Computing Δa



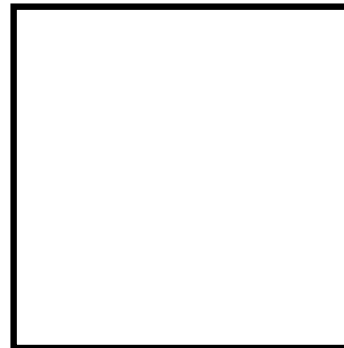
“positive” a 's
in region h



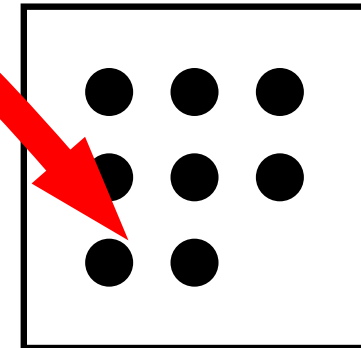
“negative” a 's
in region h

*nondeterministic
guess:
 $\#a < 0$*

*now subtract
these ones*

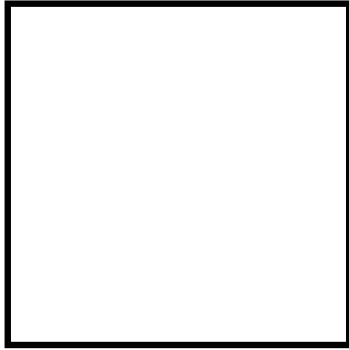


“positive”
output a 's

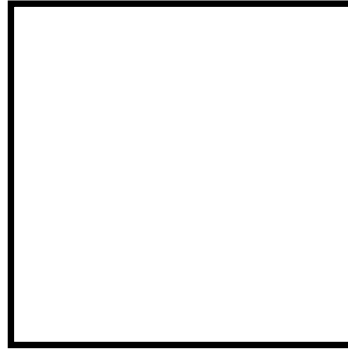


“negative”
output a 's

Computing Δa

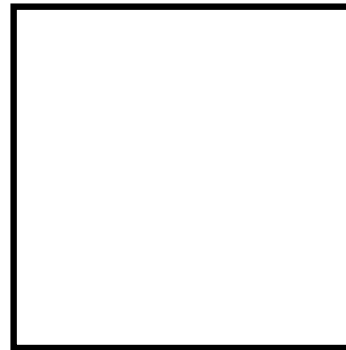


“positive” a 's
in region h

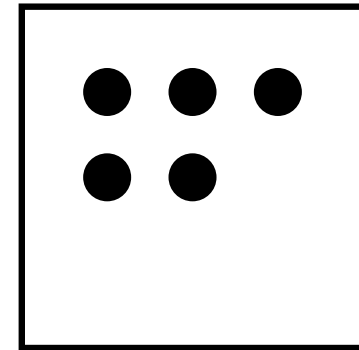


“negative” a 's
in region h

*nondeterministic
guess:
 $\#a < 0$*

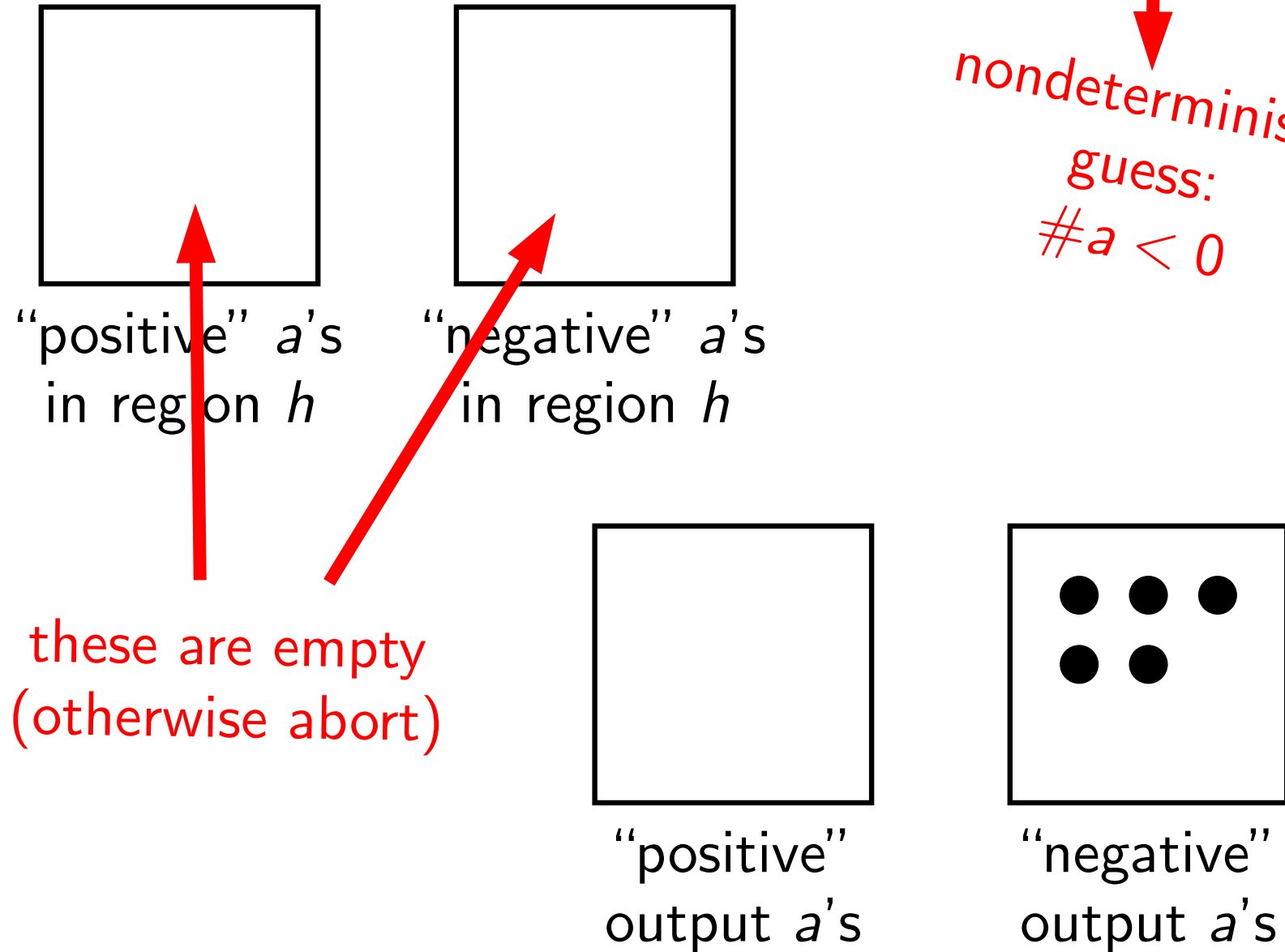


“positive”
output a 's

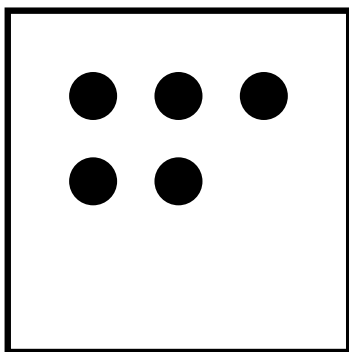


“negative”
output a 's

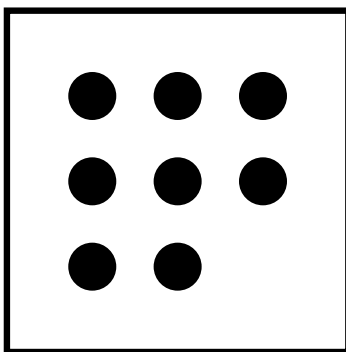
Computing Δa



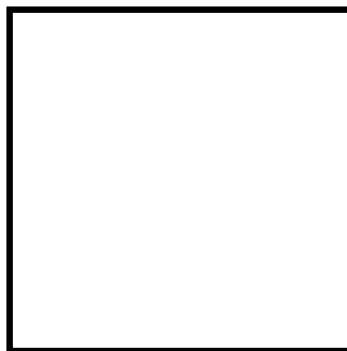
Computing Δa



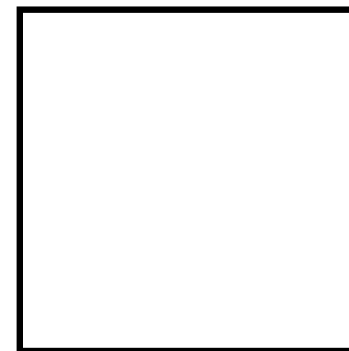
“positive” a 's
in region h



“negative” a 's
in region h

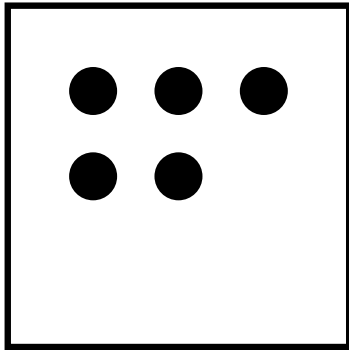


“positive”
output a 's

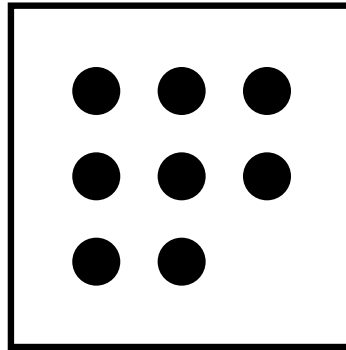


“negative”
output a 's

Computing Δa



“positive” a 's
in region h

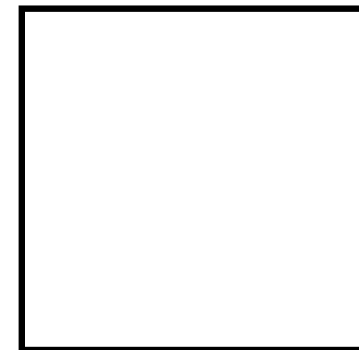


“negative” a 's
in region h

*nondeterministic
guess:
 $\#a \geq 0$*

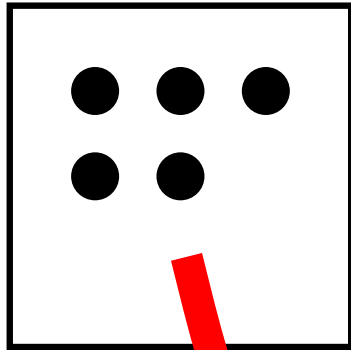


“positive”
output a 's

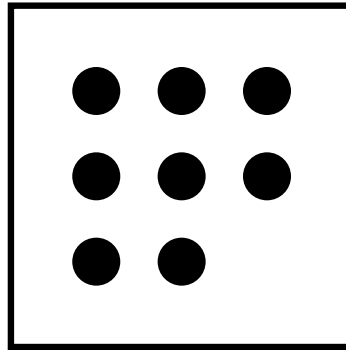


“negative”
output a 's

Computing Δa



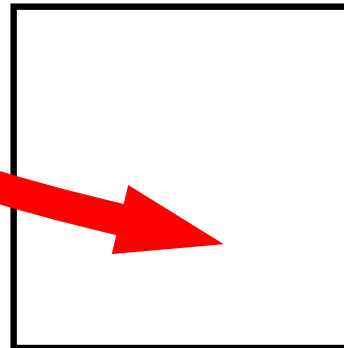
“positive” a 's
in region h



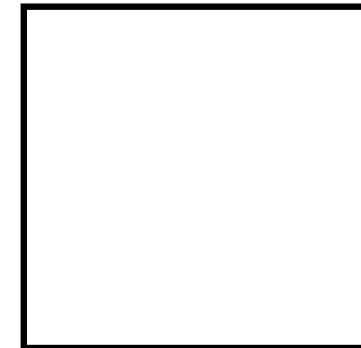
“negative” a 's
in region h

*nondeterministic
guess:
 $\#a \geq 0$*

*copy until empty
(just guess)*

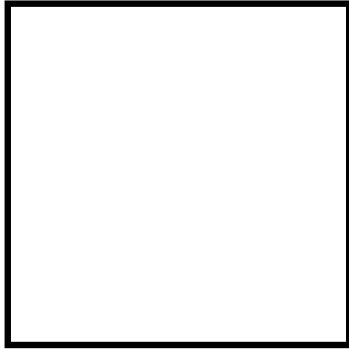


“positive”
output a 's

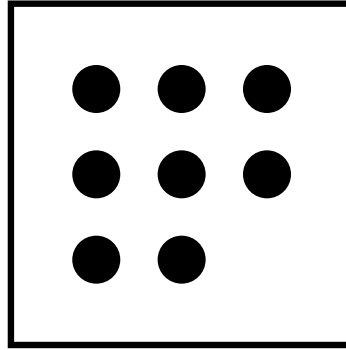


“negative”
output a 's

Computing Δa

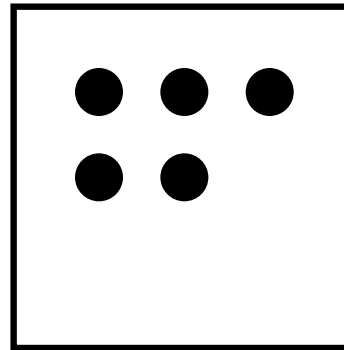


“positive” a 's
in region h

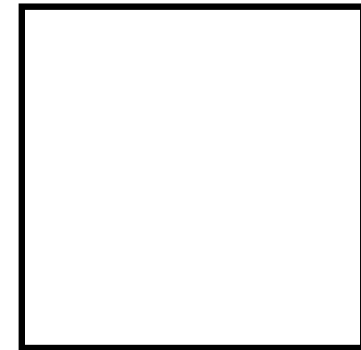


“negative” a 's
in region h

*nondeterministic
guess:
 $\#a \geq 0$*

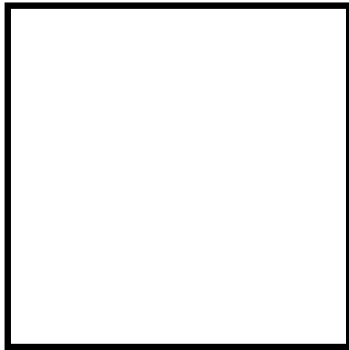


“positive”
output a 's

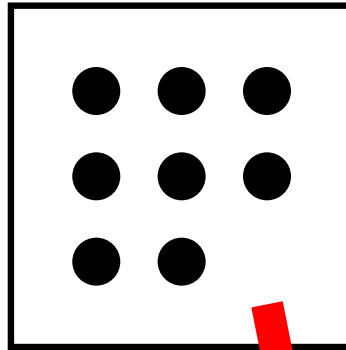


“negative”
output a 's

Computing Δa



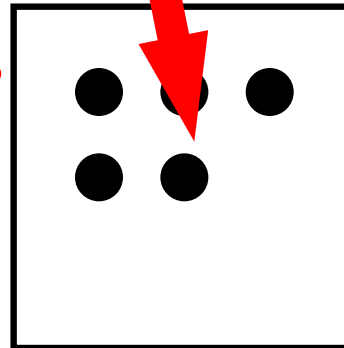
“positive” a 's
in region h



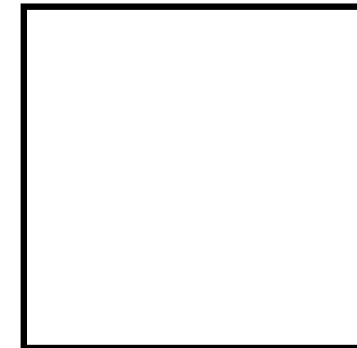
“negative” a 's
in region h

*nondeterministic
guess:
 $\#a \geq 0$*

*now subtract
these ones*

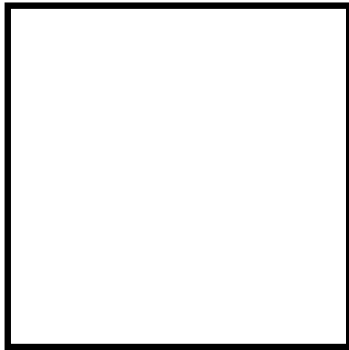


“positive”
output a 's

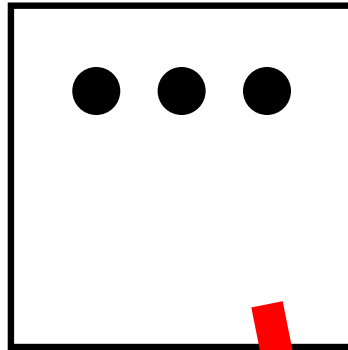


“negative”
output a 's

Computing Δa



“positive” a 's
in region h



“negative” a 's
in region h

wrong guess!



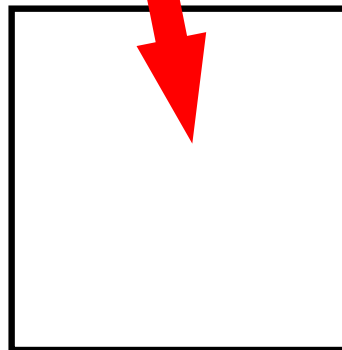
nondeterministic

guess:

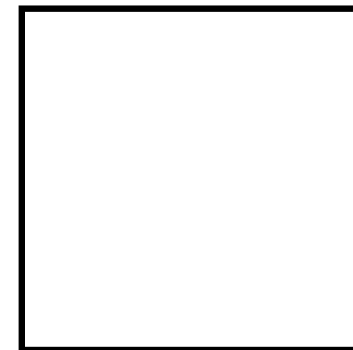
$\#a \geq 0$



FAIL



“positive”
output a 's



“negative”
output a 's

Summary

Theorem. These kinds of P system can be simulated by a partially blind register machine, and so they are not universal □

Improvement

Theorem. These kinds of P system can be simulated by a **blind** register machine, and so they are even less universal □

↑
iThanks to
Rudi & Sergiu!

¡Thanks for your attention!

¡Gracias por su atención!