Spiking Neural P Systems with Neuron Division

Jun **Wang**[1,2]
Hendrik Jan **Hoogeboom**[2]
Linqiang **Pan**[1]

[1]Huazhong University, Wuhan, China
[2]Universiteit Leiden, The Netherlands

background

efficient solutions to hard problems
eg. SAT

- nondeterminism
- exponential size resources (*precomputed*)

versus

- deterministic
- polynomial time
- polynomial size system (*initially*)

GhP: ‘linear parallel time’

‘PPP’ previous work

L. Pan, G. Păun, M.J. Pérez-Jiménez.
Spiking neural P systems with neuron division and budding.
7th Brainstorming Week on Membrane Computing, vol. II, pp. 151-168 (2009)

goal uniform solution to SAT
cf. active membranes (using spiking)

instance given as ‘input’
size of problem as parameter
precomputed: polynomial size system

here: same methodology as PPP-paper
but - do not use budding
and - constant size initial system (vs. linear)

P system

spiking neural P system with neuron division

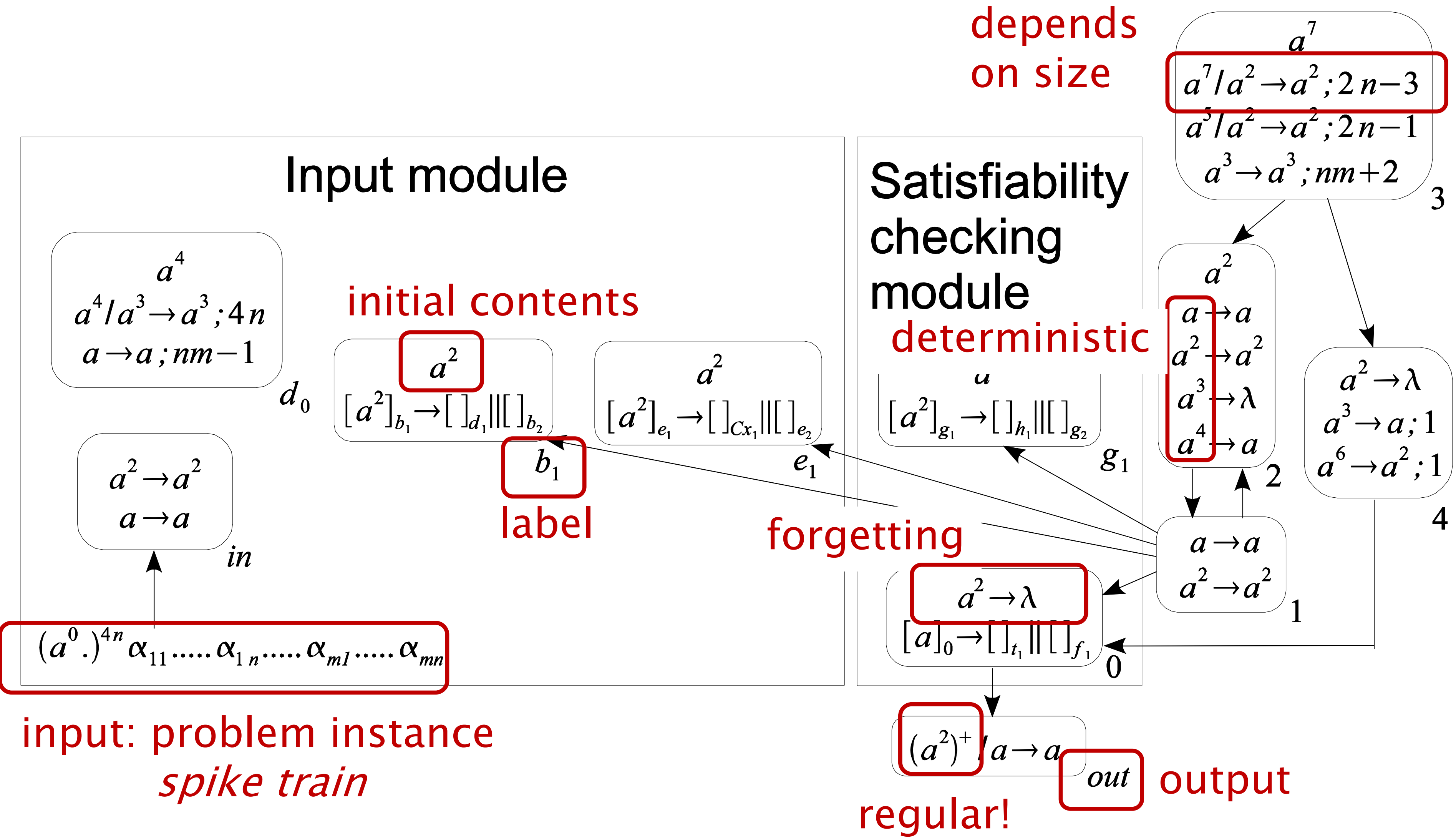
neurons	nodes	‘places’
synapses	directed edges	
spikes	objects	‘tokens’

$[\ E\ /\ a^c \rightarrow a^p\ ;\ d\ ]$ *extended firing*

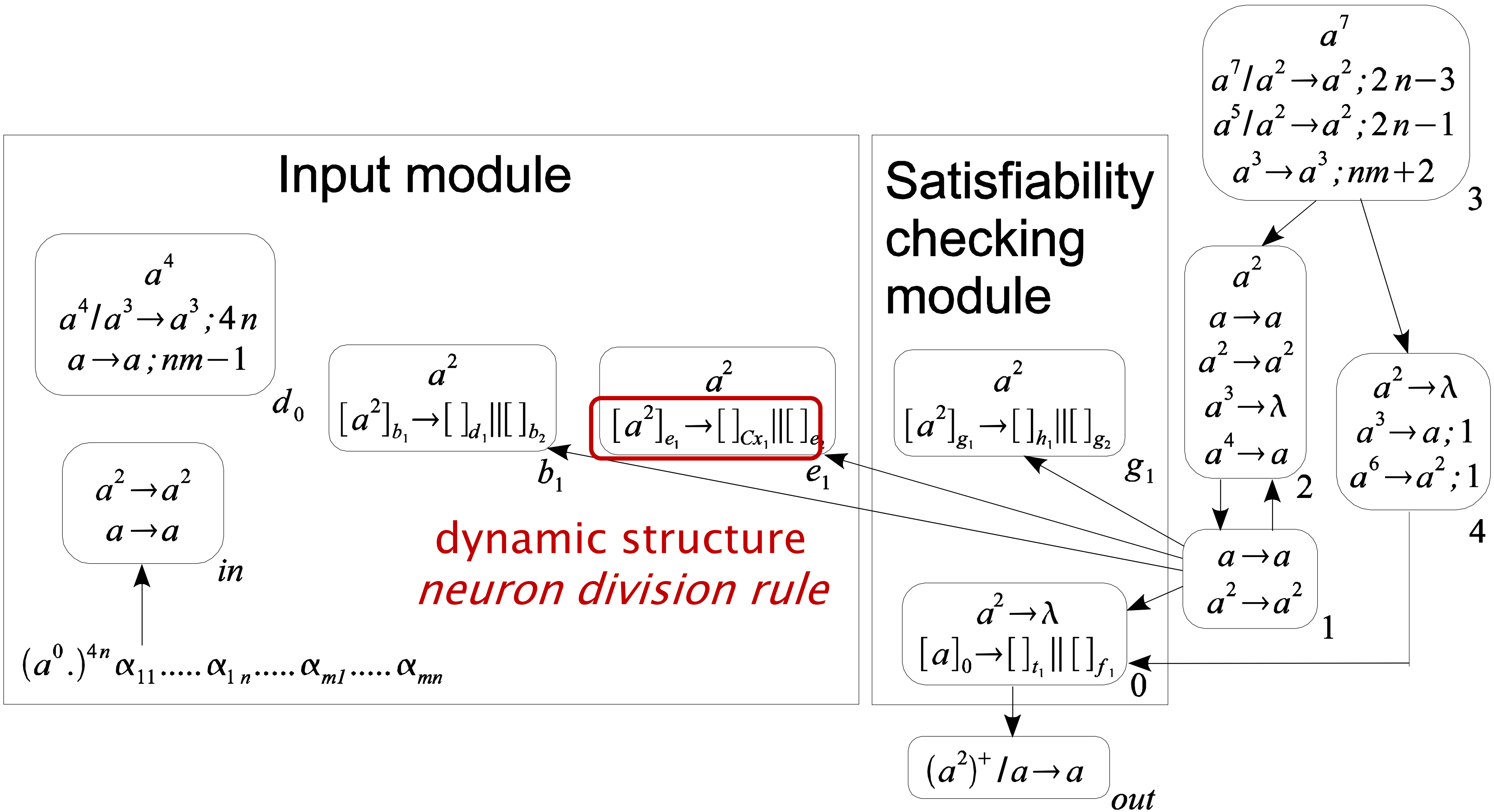
E	‘test’ regular expression over a	
$c \geq 1,\ p \geq 0,\ c \geq p$		
	consume, produce	
$d \geq 0$ *delay*		‘blocks’

parallellism – one rule for each neuron
(non)determinism

initial system



initial system



neuron division

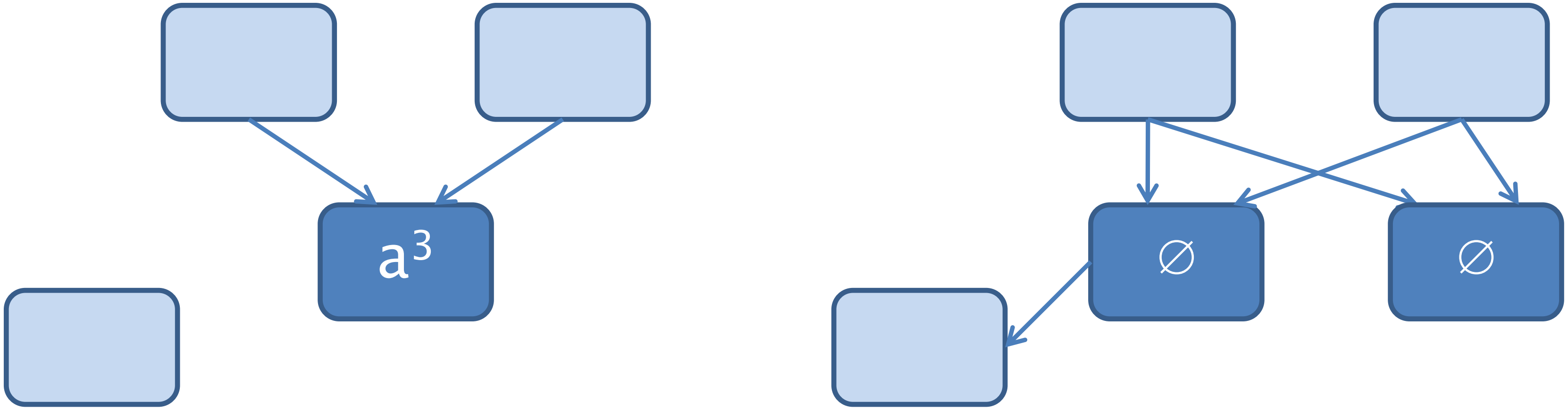
$[\ E\ ]_i \rightarrow [\ ]_j \parallel [\ ]_k$

neuron division

E regular expression 'test'

children inherit synapses
+ *synapse dictionary* (based on label)

no initial spikes



solving SAT

uniform encoding, parameters:
n variables and m clauses

SAT(n,m)

$X = \{ x_1, x_2, \ldots , x_n \}$ variables

$\gamma = C_1 \wedge C_2 \wedge \ldots \wedge C_m$ proposition
conjunction 'and' of
C_j clause,
disjunction 'or' of
x_i or $\neg x_i$ literals

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_2 \vee x_4) \wedge (\neg x_1 \vee x_3 \vee \neg x_4)$$

1 0 ? 1 0 0 ? 1

satisfiability – assignment to variables

system strategy

closely follows Pan, Păun, Pérez-Jiménez

Generation Stage

generates structure of system

division

Input Stage

reads instance γ into system

Satisfiability Checking Stage

test all valuations on clauses

Output Stage

any valuation left?

input encoding

$X = \{ x_1, x_2, \dots , x_n \}$ variables

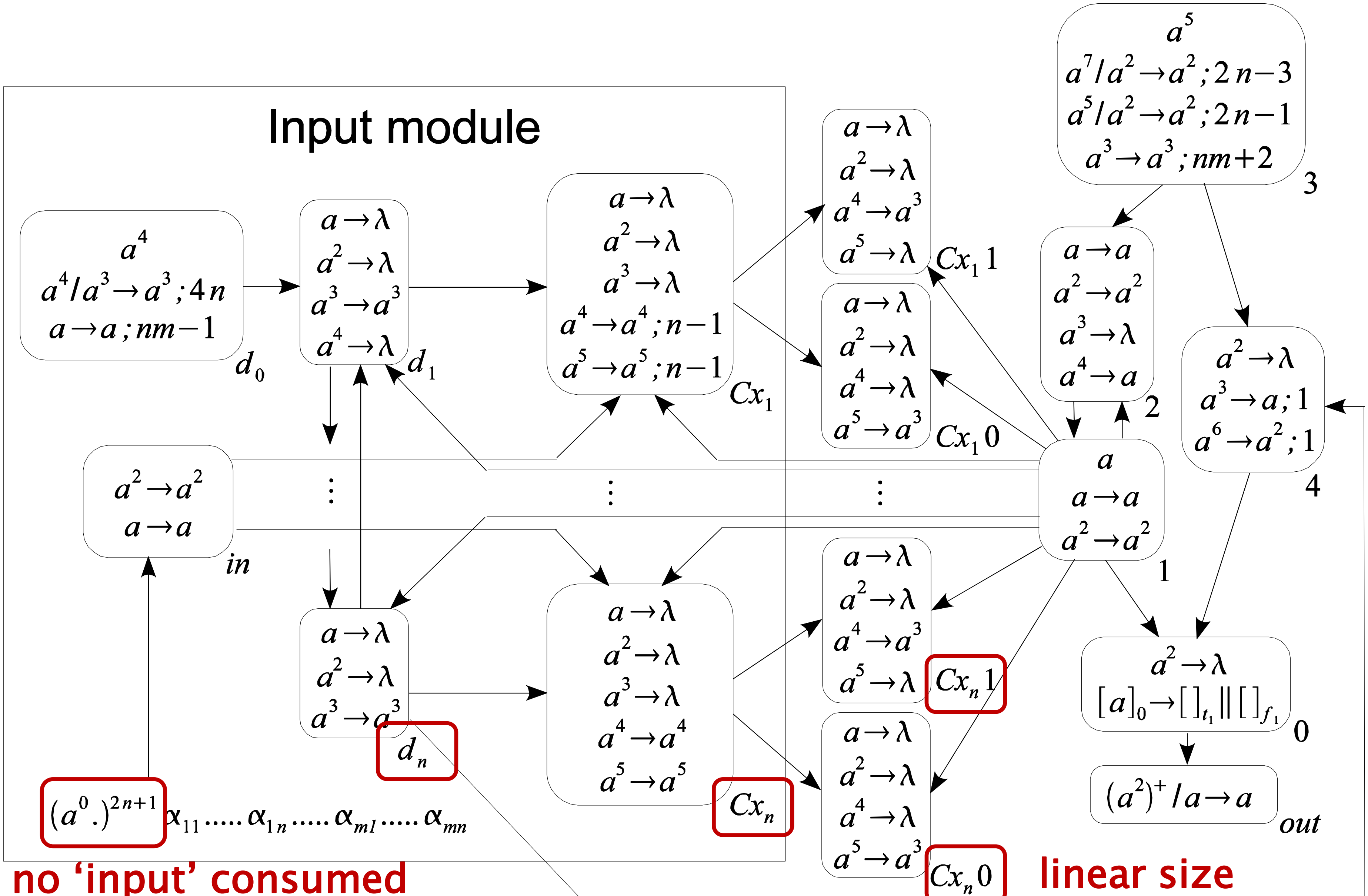
$\gamma = C_1 \wedge C_2 \wedge \dots \wedge C_m$ clauses

$$\alpha_{ij} = \begin{cases} a & x_j \text{ occurs in } C_i \\ a^2 & \neg x_j \text{ occurs in } C_i \\ a^0 & \text{otherwise} \end{cases}$$

spike train, sent into initial neuron

$$\underbrace{(a^0)^{4n}}_{\text{delay}} \alpha_{11} \dots \alpha_{1n} \dots \underbrace{\alpha_{i1}\ \alpha_{i2} \dots \alpha_{in}}_{\text{clause } C_i} \dots \alpha_{i1} \dots \alpha_{in}$$

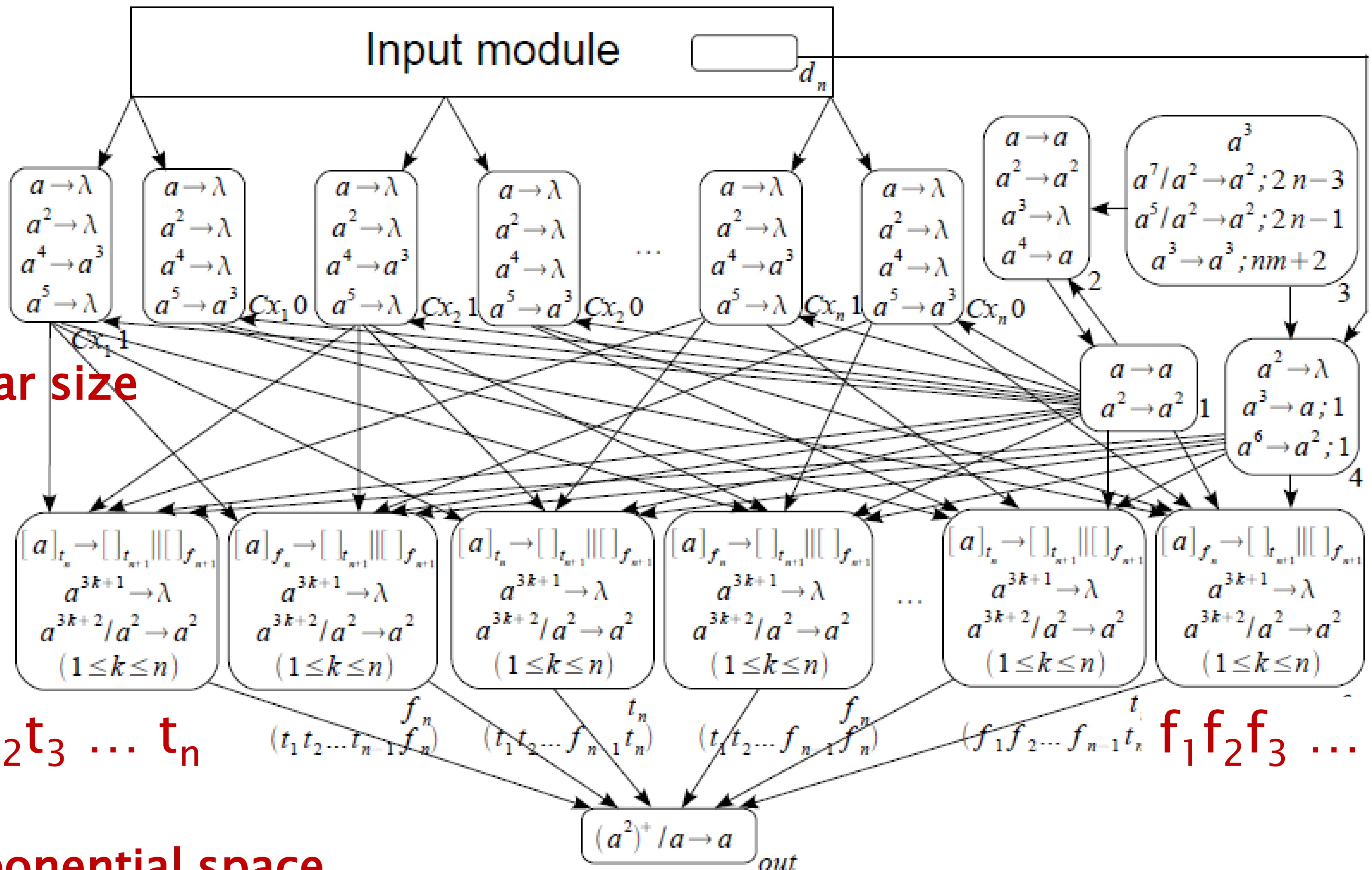
generation: step 2n−1



no ‘input’ consumed

linear size

generation: step 4n−1



linear size

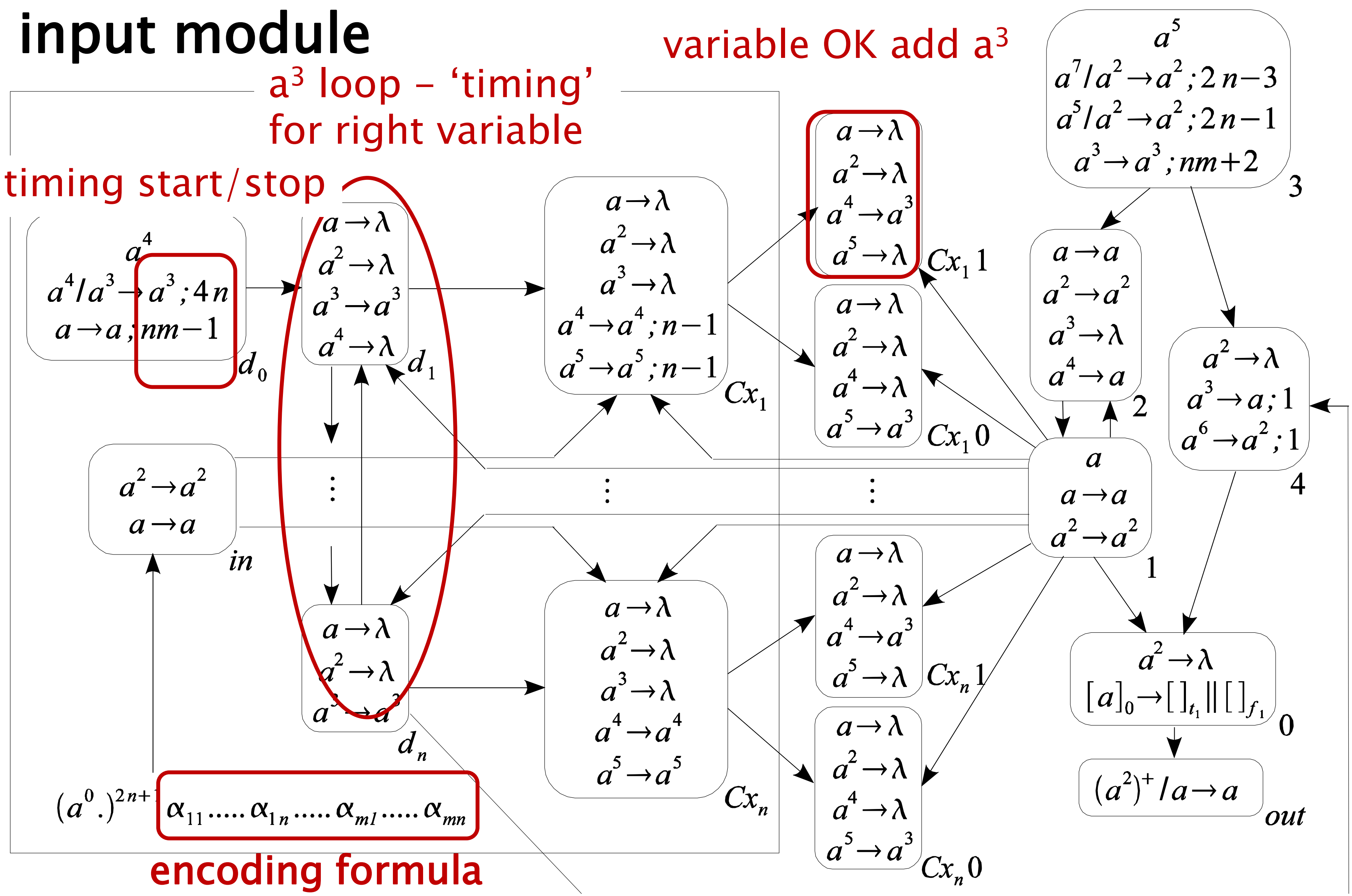
$t_1t_2t_3 \ldots t_n$

$f_1f_2f_3 \ldots f_n$

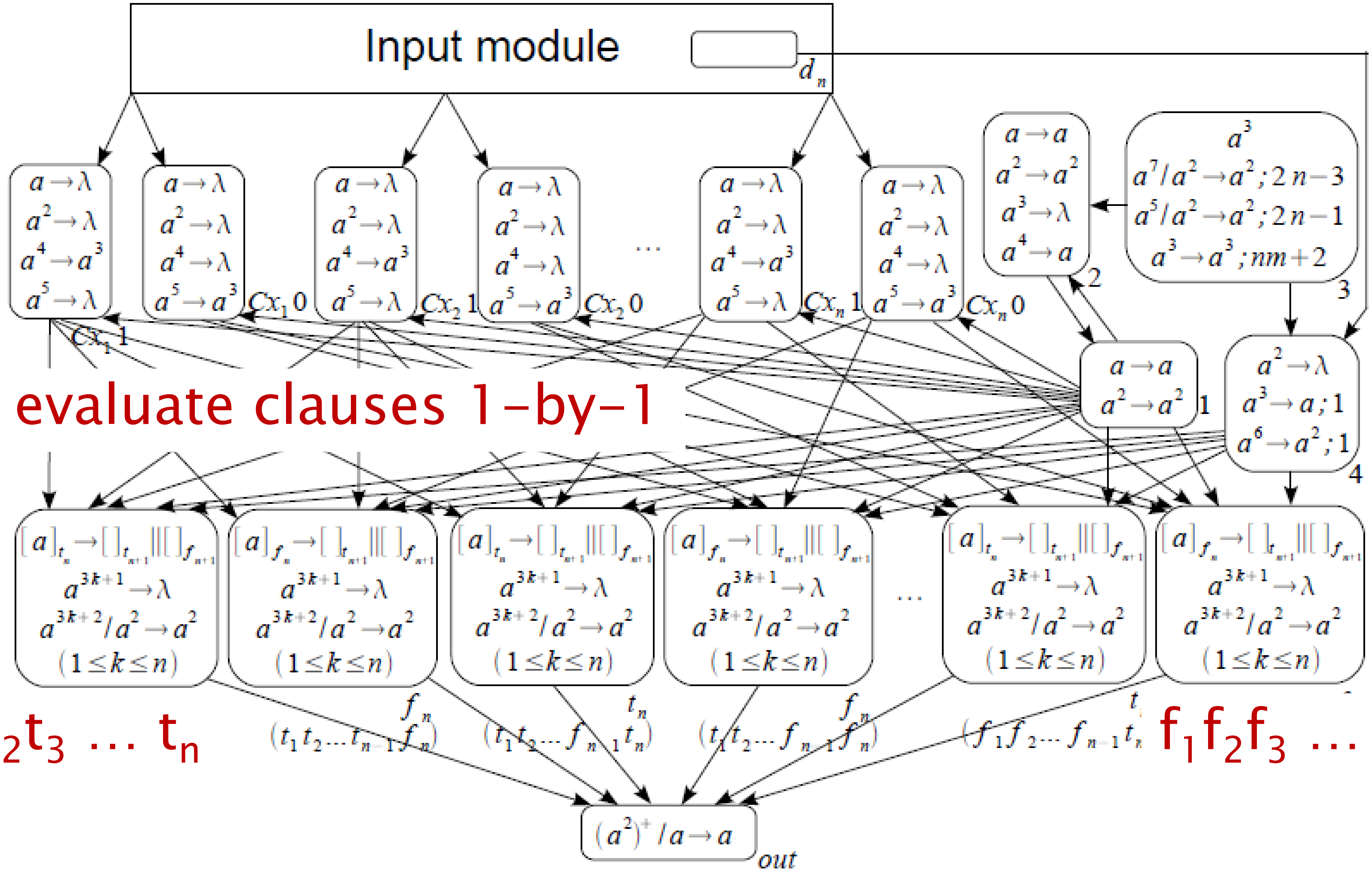
exponential space

all possible valuations

input: step 4n+1 … 4n+nm+1



satisfiability and *output*



evaluate clauses 1-by-1

$t_1 t_2 t_3 \ldots t_n$

$f_1 f_2 f_3 \ldots f_n$

split when clause fails

comparing resources

Resources	this work	“PPP”
Initial number of neurons	11	$4n + 7$
Initial number of spikes	20	9
Number of neuron labels	$10n + 7$	$6n + 8$
Size of synapse dictionary	$6n + 11$	$7n + 6$
Number of rules	$2n^2 + 26n + 26$	$n^2 + 14n + 12$

too much names/labels !
can we reuse & get right structure ?

conclusion

perhaps improved ‘timing’ can eliminate labels for consecutive ‘linear’ steps

division and/or budding

budding → linear size growth so …

initial system (PPP)