Introduction to Formal Frameworks for P-Systems

Theory Days – Day 2

2020 June 24

Ren Tristan A. de la Cruz

Membrane Computing & P Systems

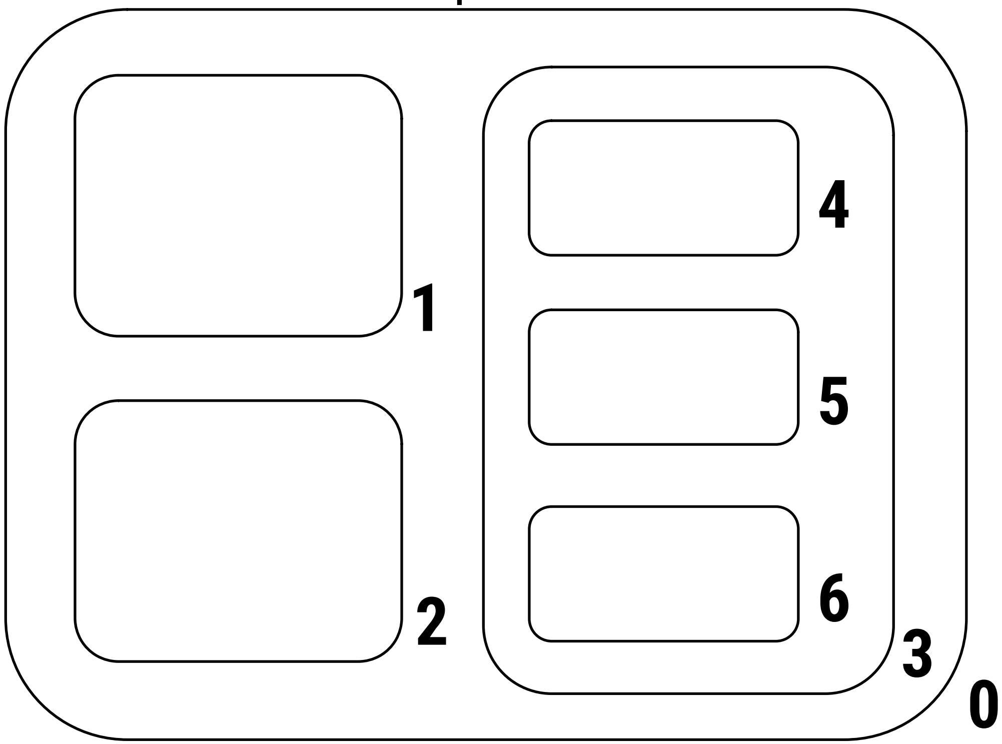
Membrane Computing – Field of Theoretical CS (Natural Computing)

Membrane computing studies different **models of computation** known as **P Systems**

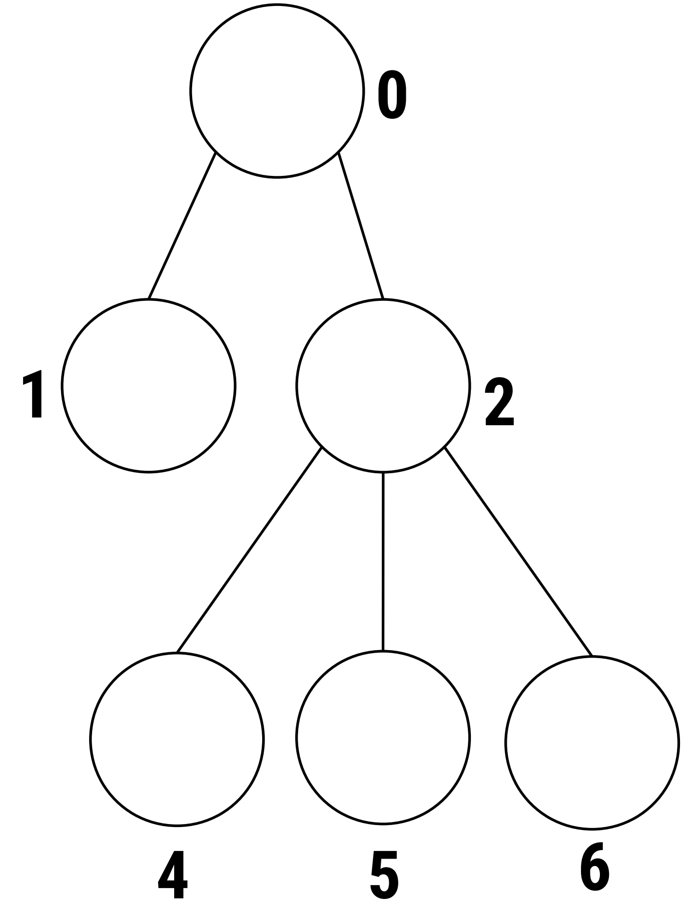
P Systems:
- distributed parallel computing devices
- biologically-inspired
- use the idea of membranes (and cells) to compartmentalize space
- related to multiset rewriting systems

P Systems – Membrane Structure

Visual Representation



Tree Representation



Balanced Brackets Representation

$$[_0 \; [_1 \,]_1 \; [_2 \,]_2 \; [_3 \; [_4 \,]_4 \; [_5 \,]_5 \; [_6 \,]_6 \;]_3 \;]_0$$

[1] *Păun*, G.. 1998. *Computing with Membranes*. In *Technical Report. Turku Centre for Computer Science*.

P Systems – Multisets of Objects

Alphabet / Set of Objects $\mathbf{O} = \{a,b,c\}$

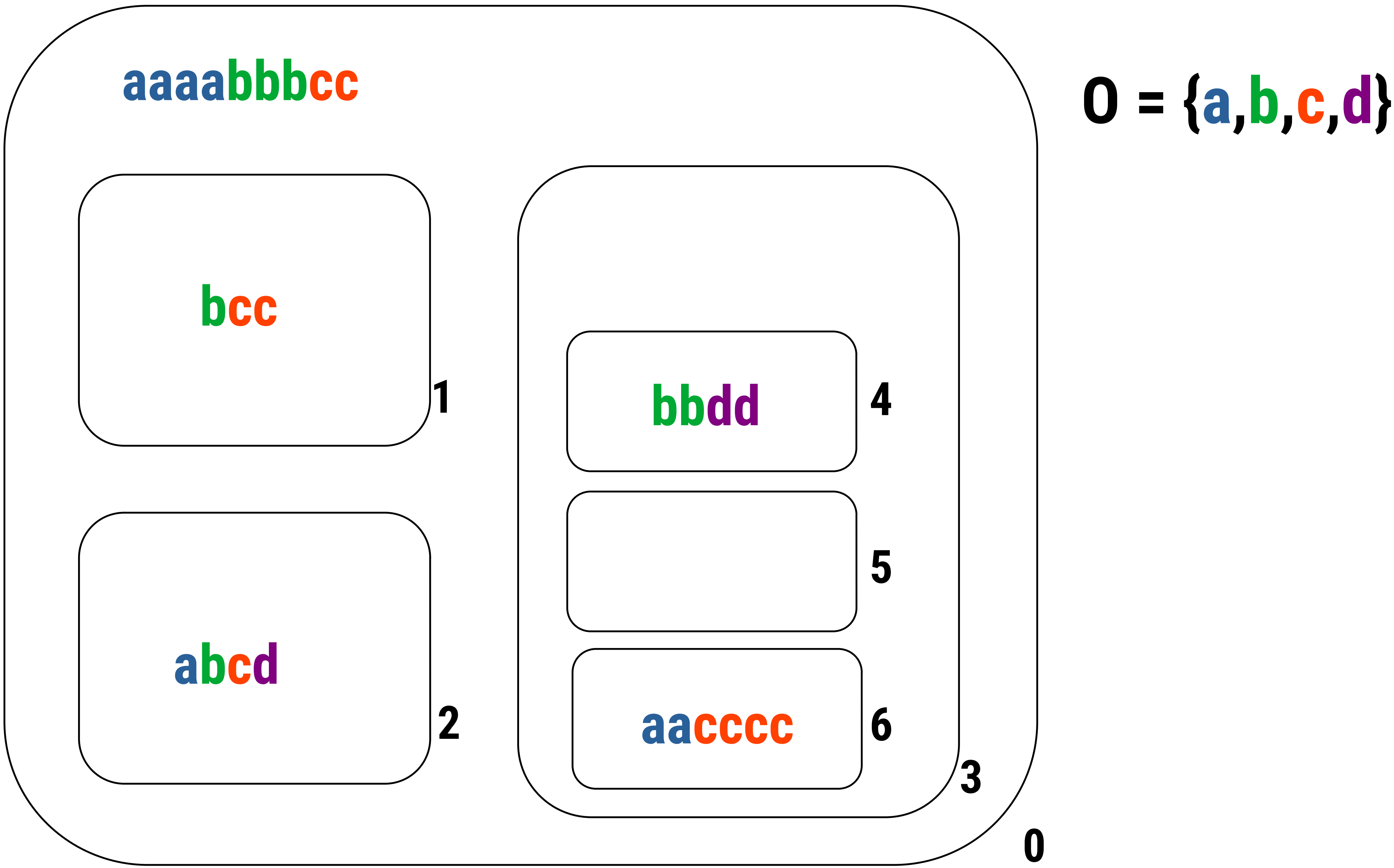
Sample Multisets over **O**

$aaaabbcc = \{a,a,a,a,b,b,c,c\} = a^4b^2c^2$

$abbccc = \{a,b,b,c,c,c\} = a^1b^2c^3$

$ac = \{a,c\} = a^1c^1 = a^1b^0c^1$

P Systems – Multisets in Membranes



P Systems – Multiset Rewriting

Alphabet / Set of Objects **O = {a,b,c}**

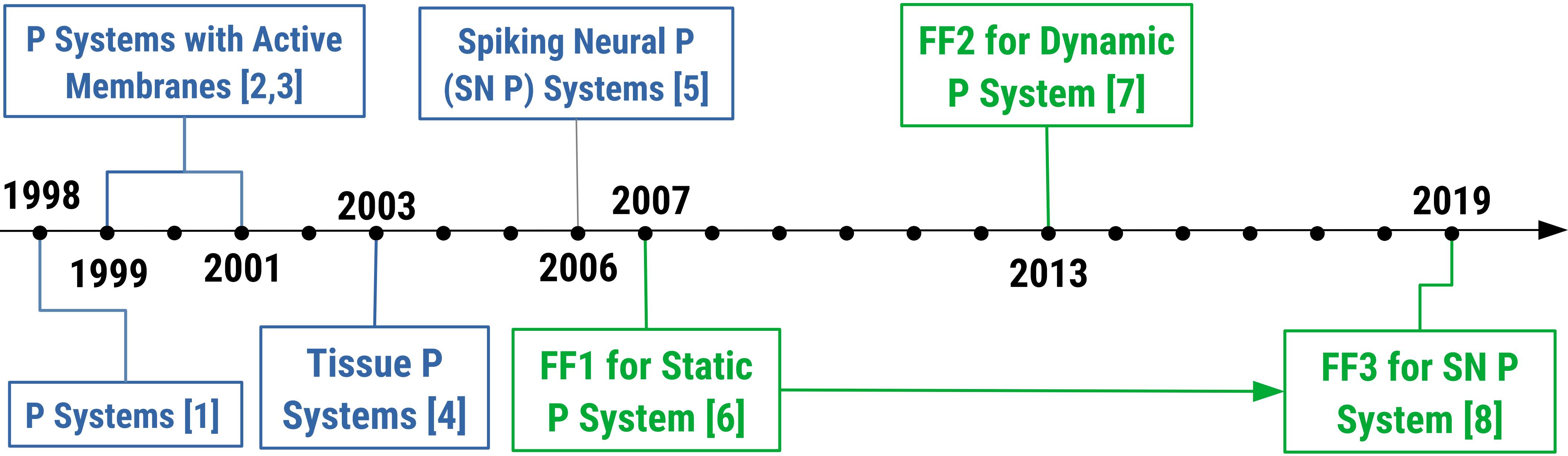
aaaabbcc	bc → aaa	a^7bc
acc	ac → bb	bbc
aabbbccc	ab → aab	aaabbbccc

P Systems – Multiset Rewriting

Alphabet / Set of Objects **O = {a,b,c}**

aaaabbcc	bc	$\xrightarrow{x2}$	aaa	a^{10}
acc	ac	$\rightarrow$	bb	bbc
aabbbccc	ab	$\xrightarrow[x2]{}$	aab	aaaabbbccc

P Systems and Formal Frameworks (FFs)



[1] *Păun*, G.. 1998. *Computing with Membranes*. In *Technical Report. Turku Centre for Computer Science*.

[2] *Păun*, G.. 1999. *P Systems with Active Membranes: Attacking NP-Complete Problems*. In *Centre for Discrete Mathematics and Theoretical Computer Science* (CDMTCS-102) – Research Report Series

[3] *Păun*, G.. 2001. *P Systems with Active Membranes: Attacking NP-Complete Problems*. In *Journal of Automata, Languags, and Combinatorics*. vol.6, issue 1 (January 2001), 75-90.

[4] *Martín-Videa*, C., *Păun*, G., *Pazos*, J., *Rodríguez-Patón*, A.. 2003. *Tissue P Systems*. In *Theoretical Computer Science*. Elsevier.

[5] *Ionescu*, M., Păun, G., *Yokomori*, T.. 2006. *Spiking Neural P Systems*. In *Fundamenta Informaticae*. vol 71, issue 2,3 (February 2006), 279-308.

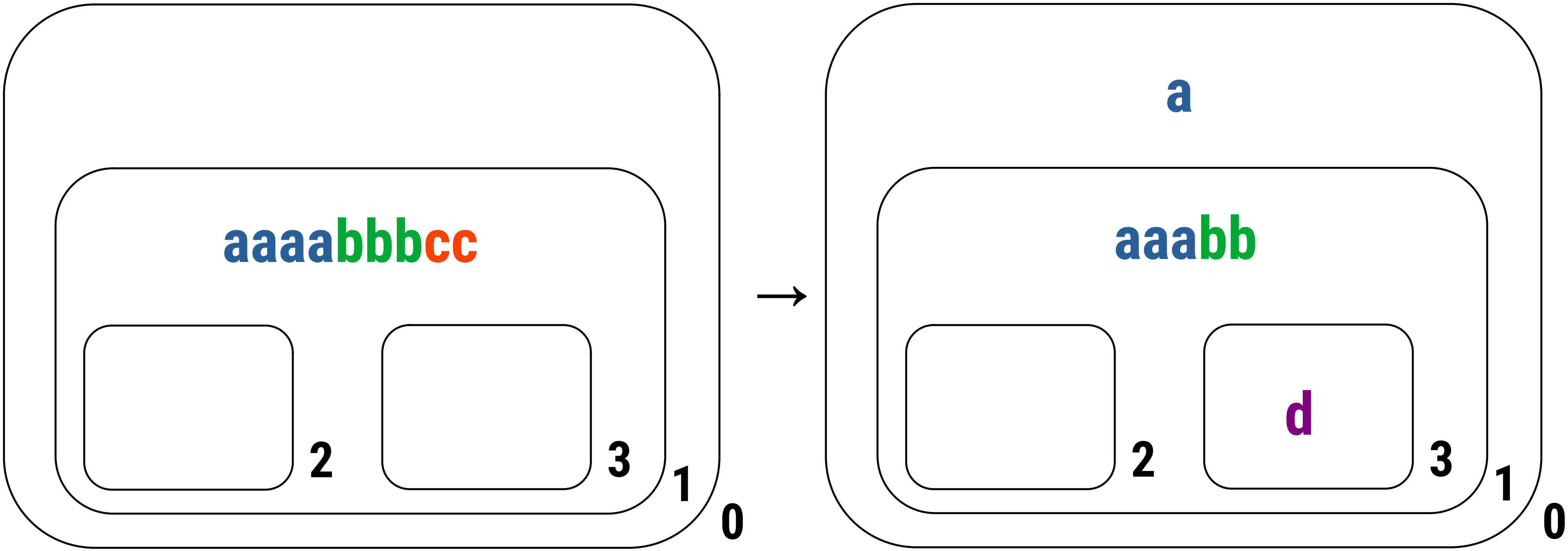
[6] *Freund* R., *Verlan* S. 2007. *A Formal Framework for Static (Tissue) P Systems*. In: Eleftherakis G., Kefalas P., Păun G., Rozenberg G., Salomaa A. (eds) *Membrane Computing*. WMC 2007. Lecture Notes in Computer Science, vol 4860. Springer, Berlin, Heidelberg

[7] *Freund*, R., *Pérez-Hurtado*, I., *Riscos-Núñez*, A., *Verlan*, S.. 2013. *A Formalization of Membrane Systems with Dynamically Evolving Structures.* In *International Journal of Computer Mathematics*, 90:4, 801-815

[8] *Verlan*, S., *Freund*, R., *Alhazov*, A., *Pan*, L.. 2008. A Formal Framework for Spiking Neural P Systems. In *Proceedings of 20th Conference on Membrane Computing (CMC20)*

P Systems

$O = \{a, b, c, d\}$



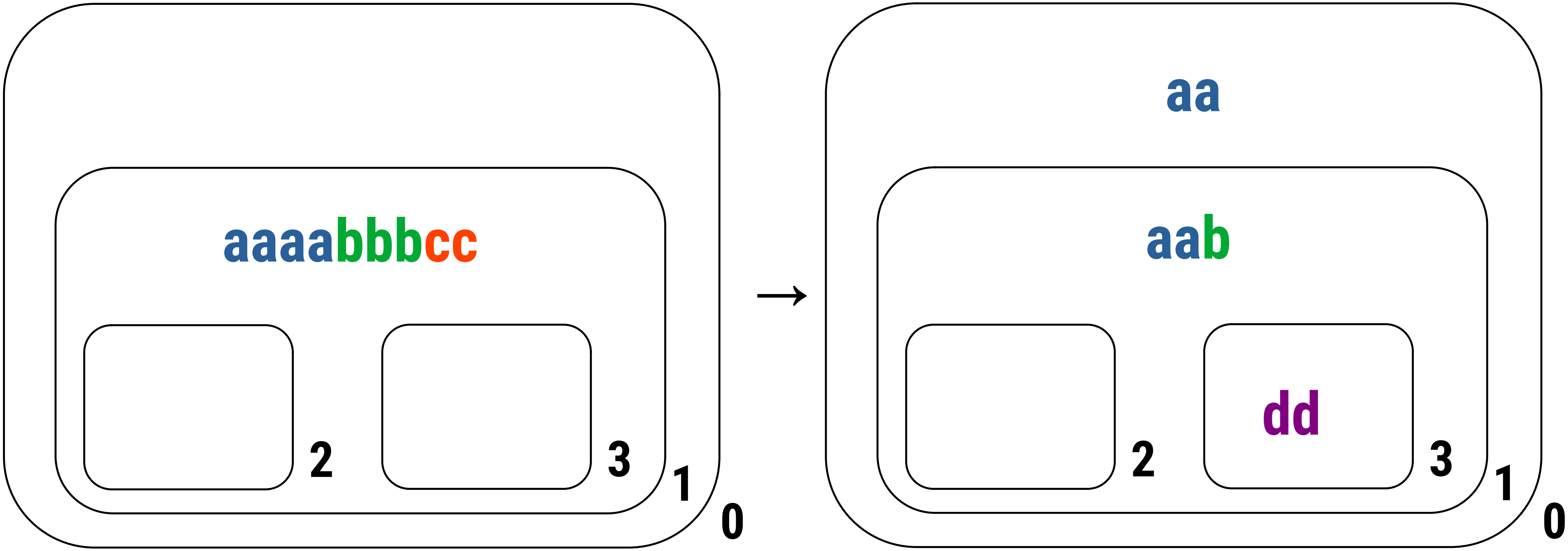
Rule in Membrane 1: $abcc \rightarrow (a, \mathit{out})(d, \mathit{in}_3)$

$$[abcc]_1 \rightarrow [a]_0 [d]_3$$

[1] *Păun, G..* 1998. *Computing with Membranes*. In *Technical Report. Turku Centre for Computer Science*.

P Systems

$O = \{a, b, c, d\}$



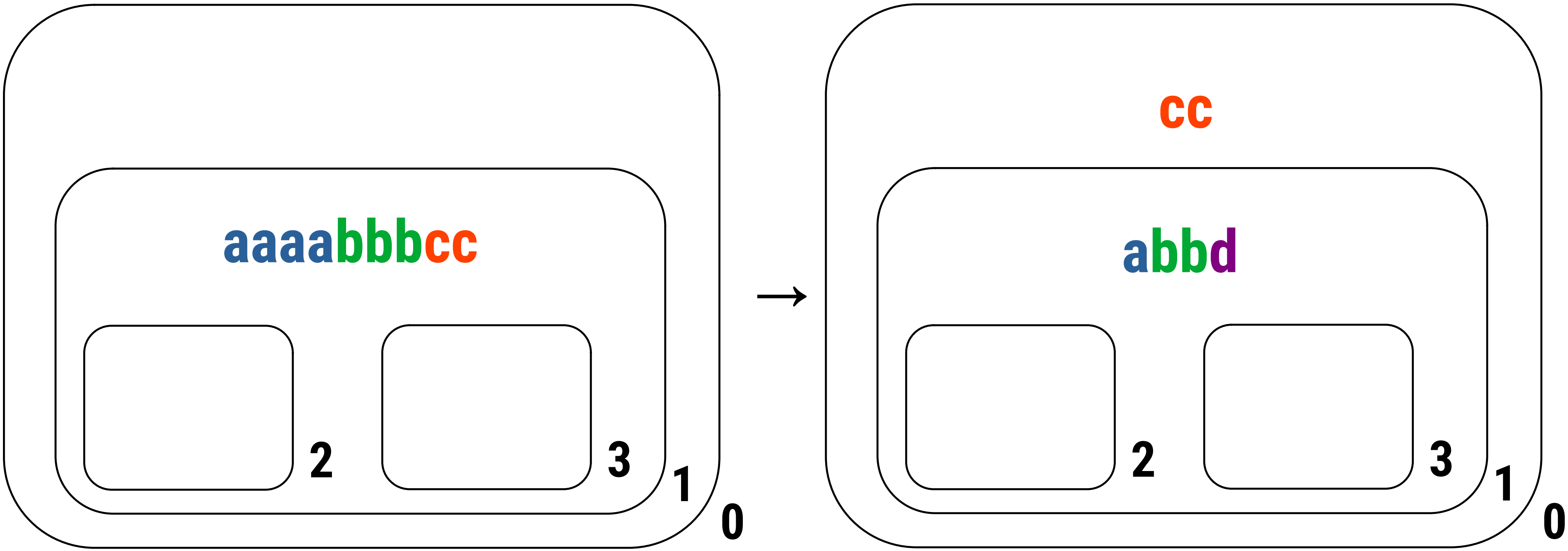
Rule in Membrane 1: $abc \rightarrow (a, out)(d, in_3)$ x2

$[abc]_1 \rightarrow [a]_0 [d]_3$ x2

[1] *Păun*, G.. 1998. *Computing with Membranes*. In *Technical Report. Turku Centre for Computer Science*.

P Systems

$O = \{a,b,c,d\}$



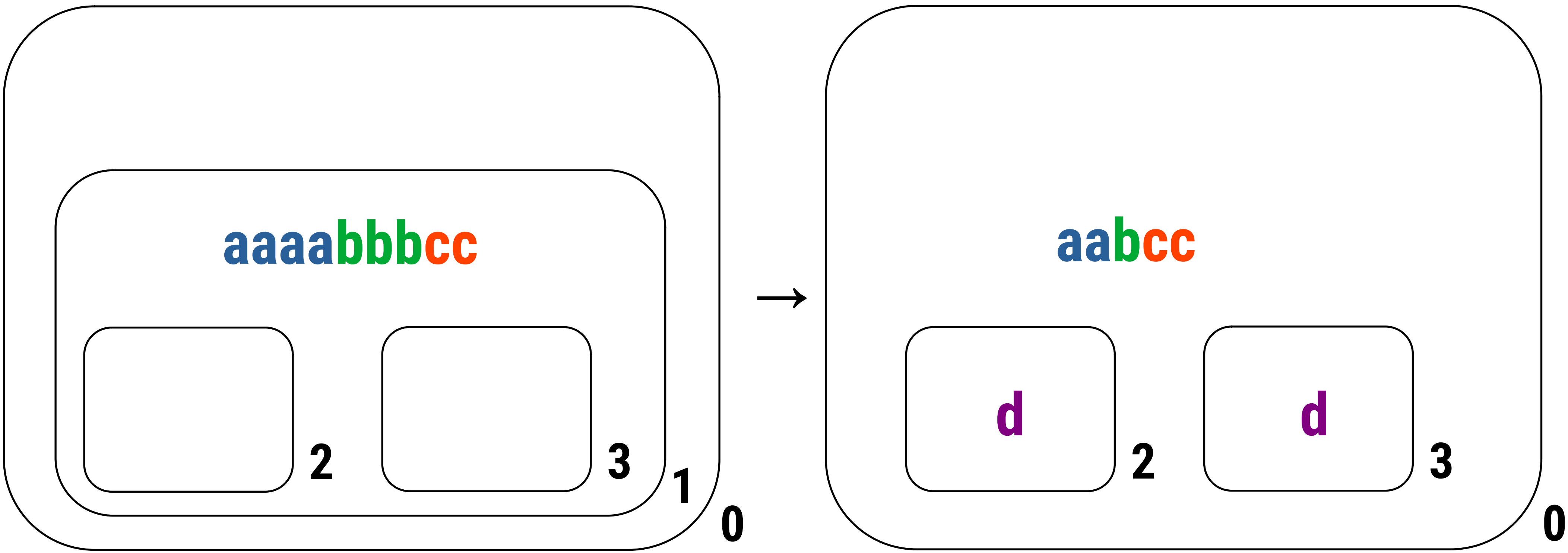
Rule in Membrane 1: $aaabcc \rightarrow (cc, \mathit{out})(d, \mathit{here})$

$$[aabcc]_1 \rightarrow [cc]_0 [d]_1$$

[1] *Păun*, G.. 1998. *Computing with Membranes*. In *Technical Report. Turku Centre for Computer Science*.

P Systems

$O = \{a, b, c, d\}$



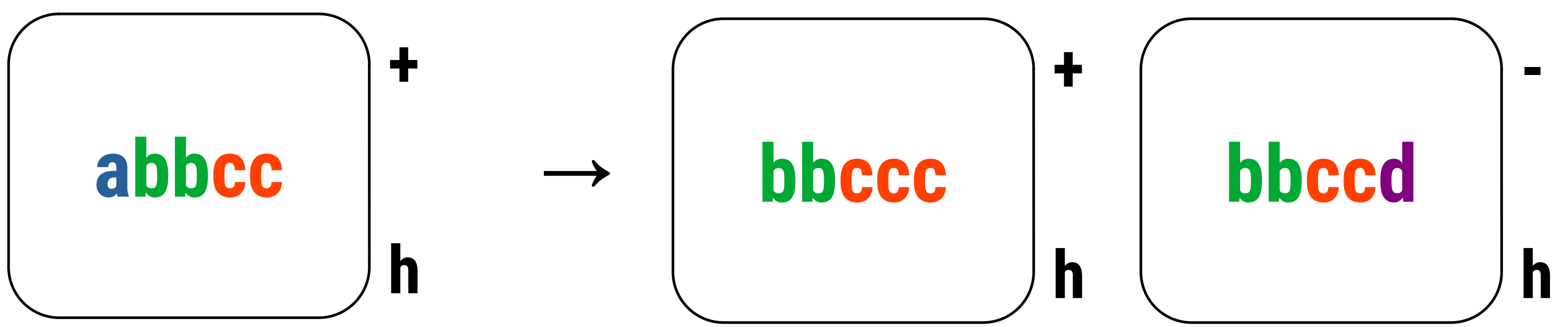
Rule in Membrane 1: $aabbcc \rightarrow (d, in_2)(d, in_3)\delta$

$$[aabbcc]_1 \rightarrow [d]_2\,[d]_3\,[\delta]_1$$

[1] *Păun*, G.. 1998. *Computing with Membranes*. In *Technical Report. Turku Centre for Computer Science*.

P Systems with Active Membranes

$O = \{a,b,c,d\}$ $P = \{+,-,0\}$



$$[a]_h^+ \rightarrow [c]_h^+ [d]_h^-$$

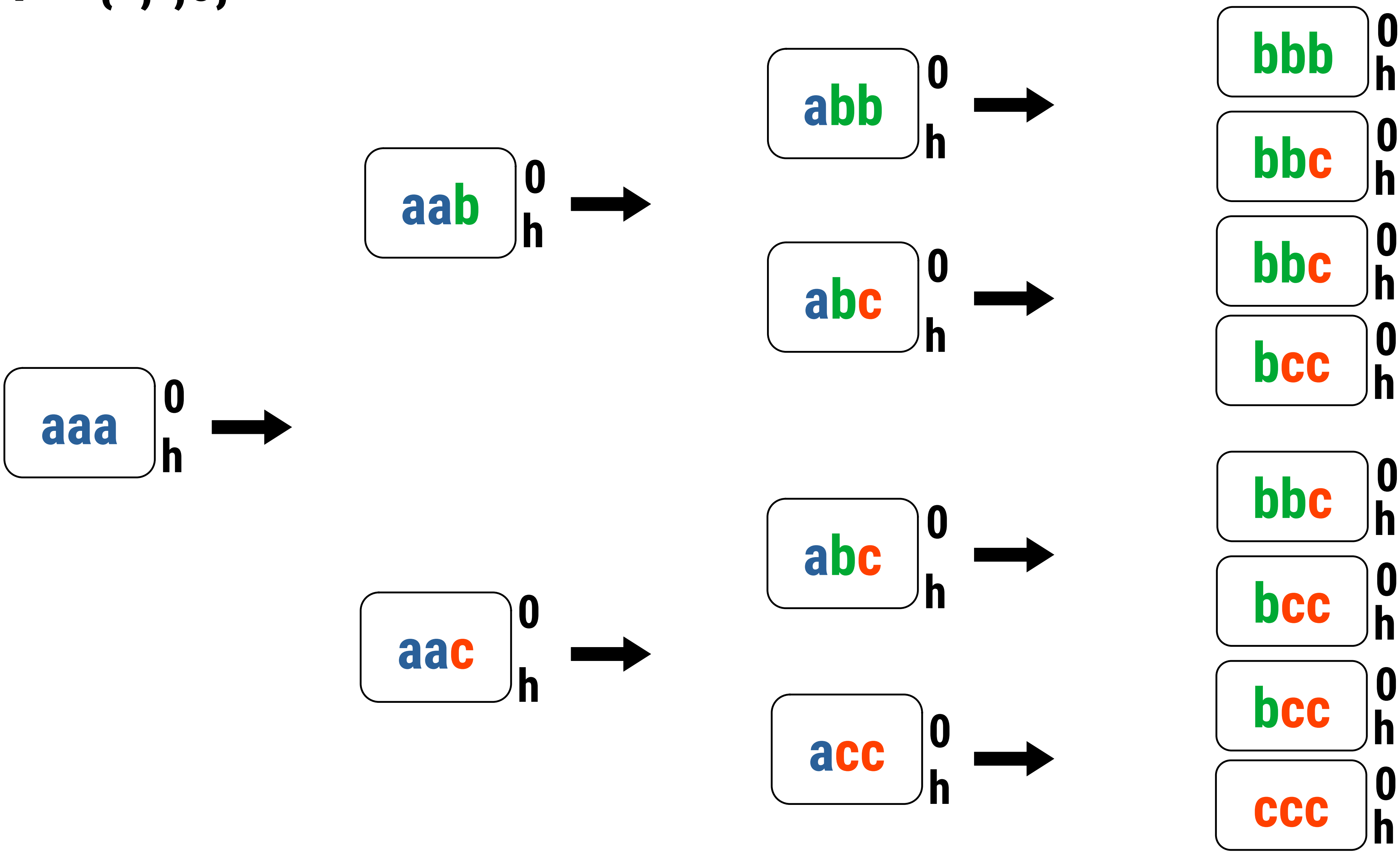
[2] *Păun*, G.. 1999. *P Systems with Active Membranes: Attacking NP-Complete Problems*. In *Centre for Discrete Mathematics and Theoretical Computer Science* (CDMTCS-102) – Research Report Series

[3] *Păun*, G.. 2001. *P Systems with Active Membranes: Attacking NP-Complete Problems*. In *Journal of Automata, Languags, and Combinatorics*. vol.6, issue 1 (January 2001), 75-90.

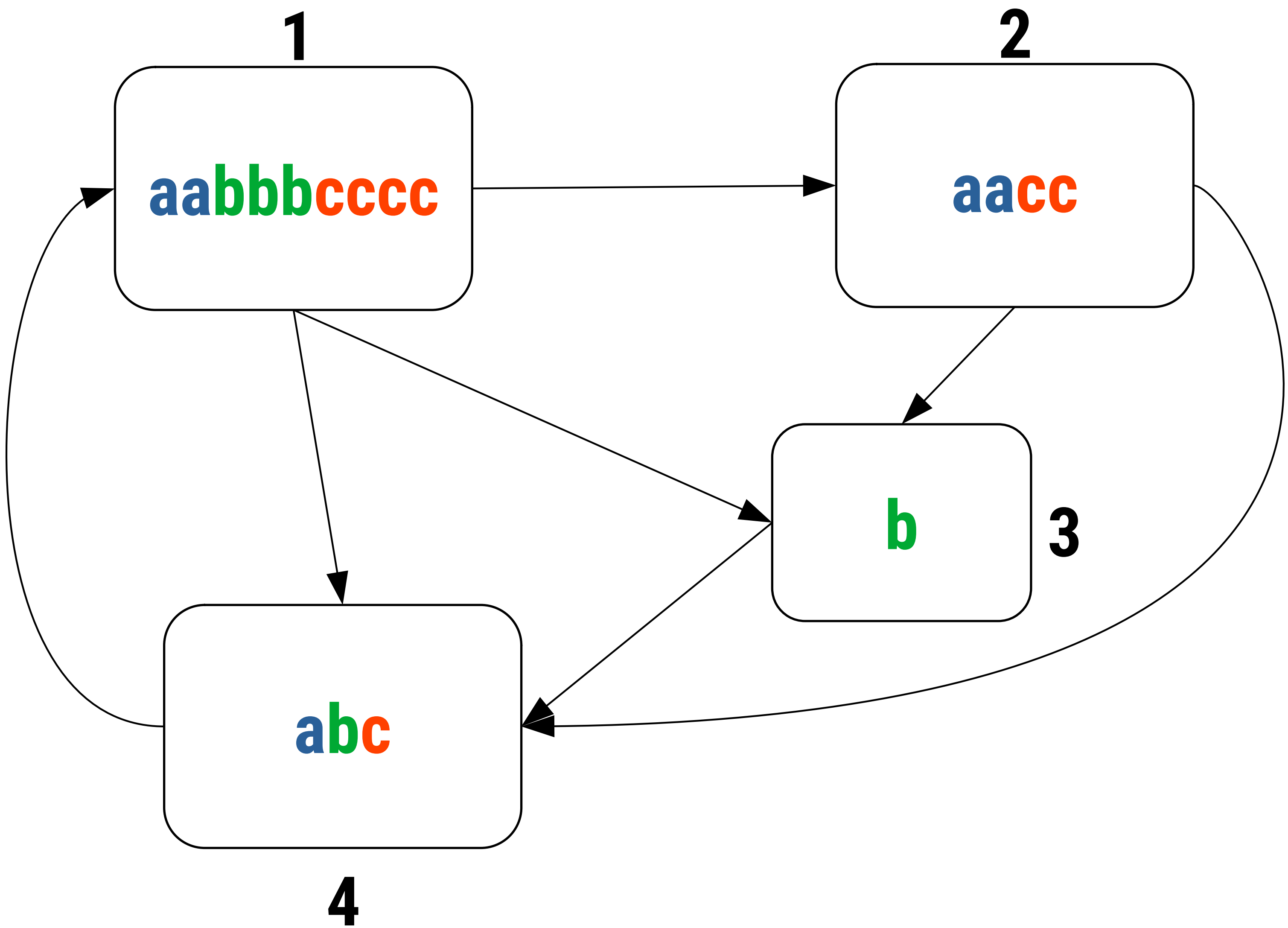
P Systems with Active Membranes

$O = \{a, b, c\}$ $\quad [a]_h^0 \rightarrow [b]_h^0\ [c]_h^0$

$P = \{+, -, 0\}$

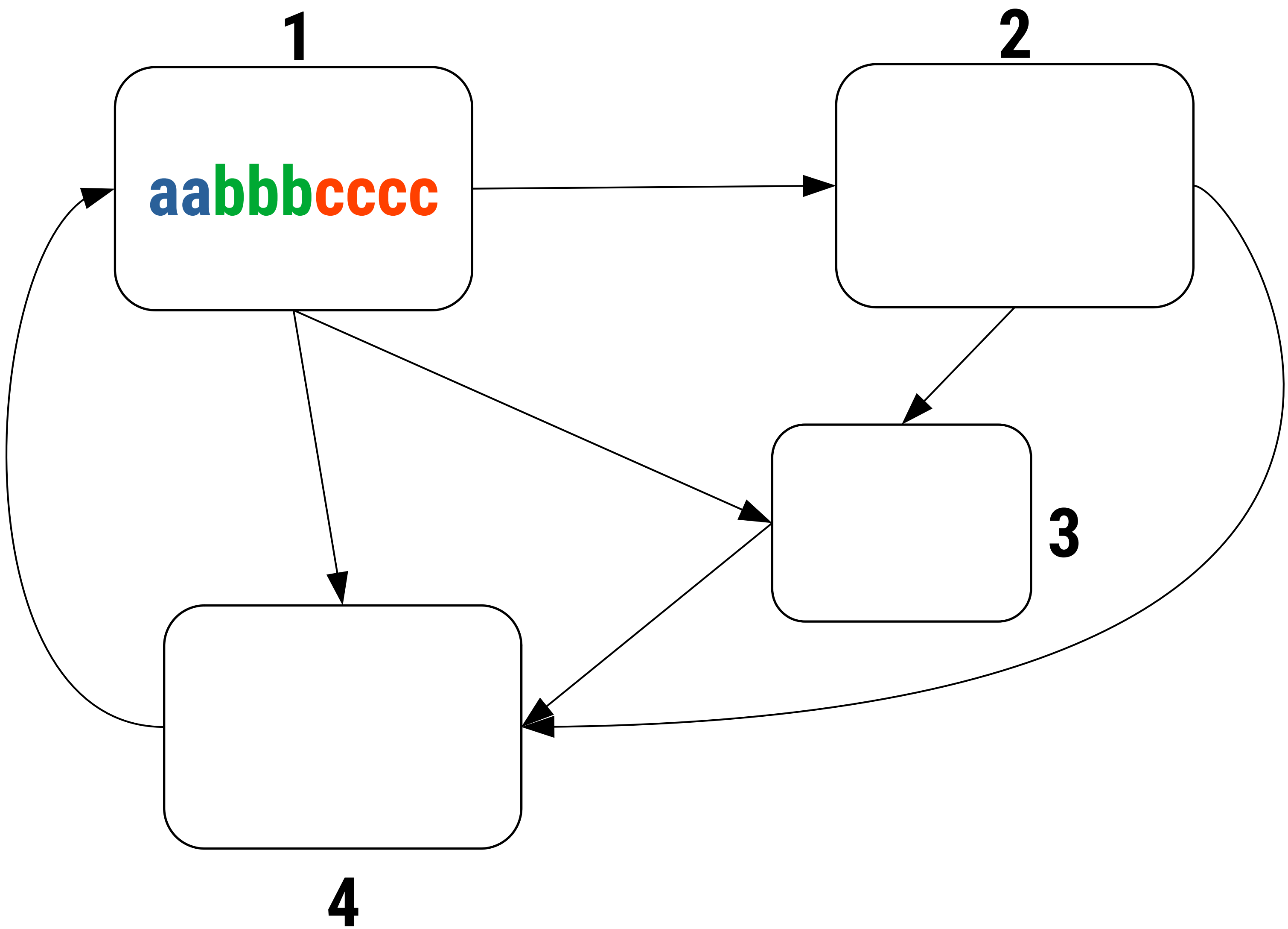


(Static) Tissue P Systems



[4] *Martín-Videa*, C., *Păun*, G., *Pazos*, J., *Rodríguez-Patón*, A.. 2003. *Tissue P Systems*. In *Theoretical Computer Science.* Elsevier.

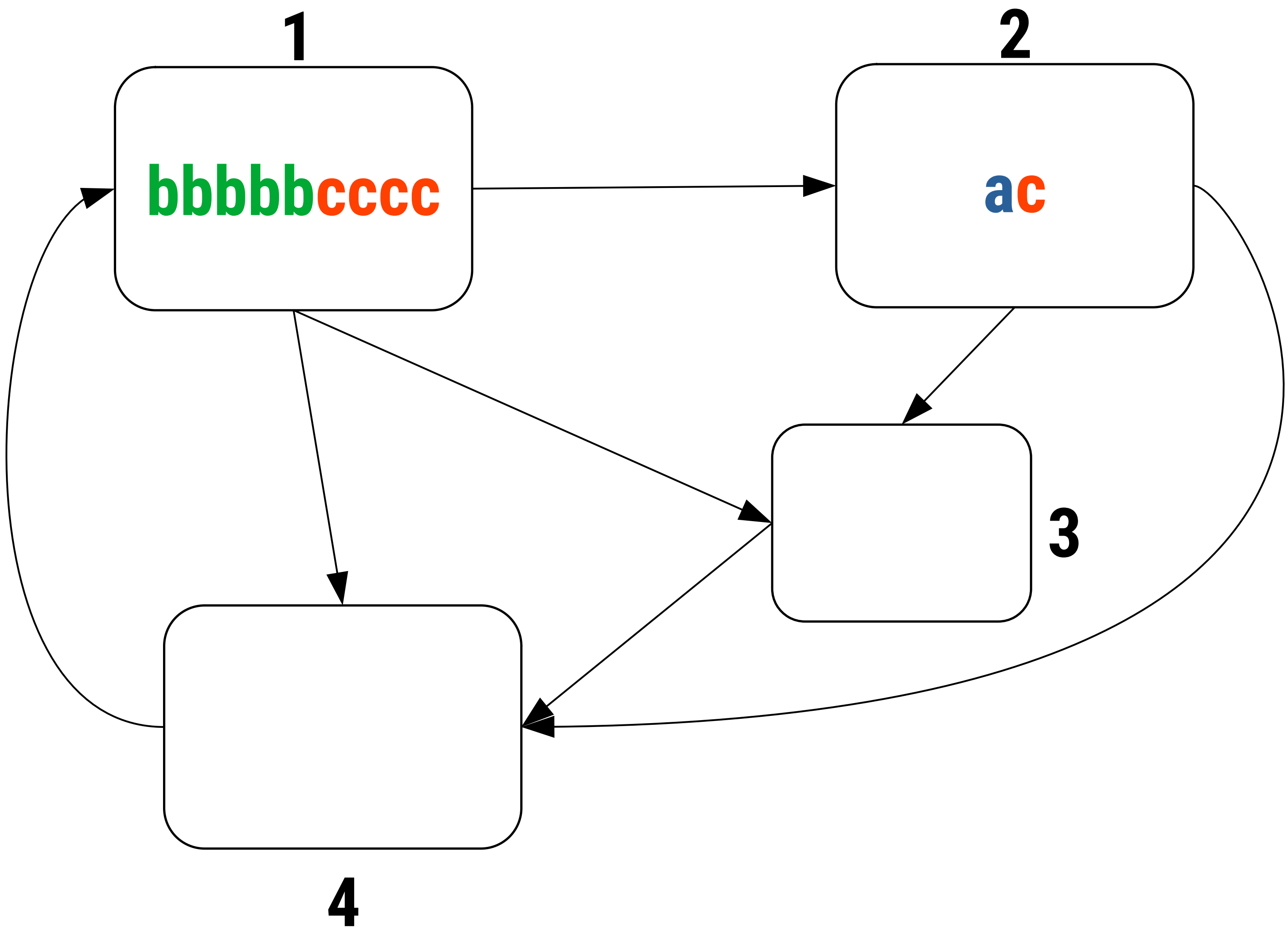
(Static) Tissue P Systems



Rule in Cell 1: aa→(bb, *here*)(ac, go)

[4] *Martín-Videa*, C., *Păun*, G., *Pazos*, J., *Rodríguez-Patón*, A.. 2003. *Tissue P Systems*. In *Theoretical Computer Science.* Elsevier.

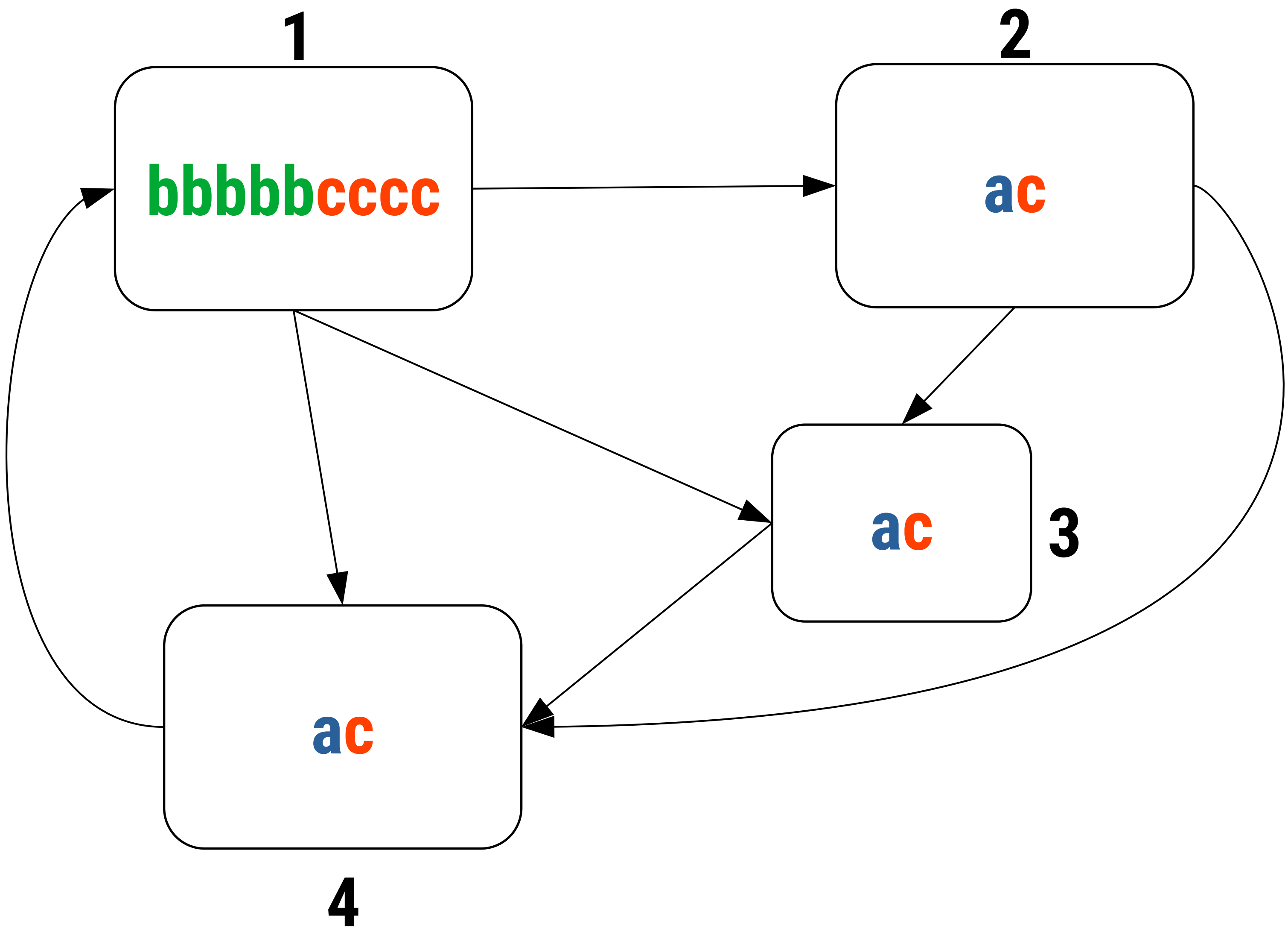
(Static) Tissue P Systems



Rule in Cell 1: aa→(bb, *here*)(ac, go)

[4] *Martín-Videa*, C., *Păun*, G., *Pazos*, J., *Rodríguez-Patón*, A.. 2003. *Tissue P Systems*. In *Theoretical Computer Science.* Elsevier.

(Static) Tissue P Systems



Rule in Cell 1: aa→(bb, *here*)(ac, go)

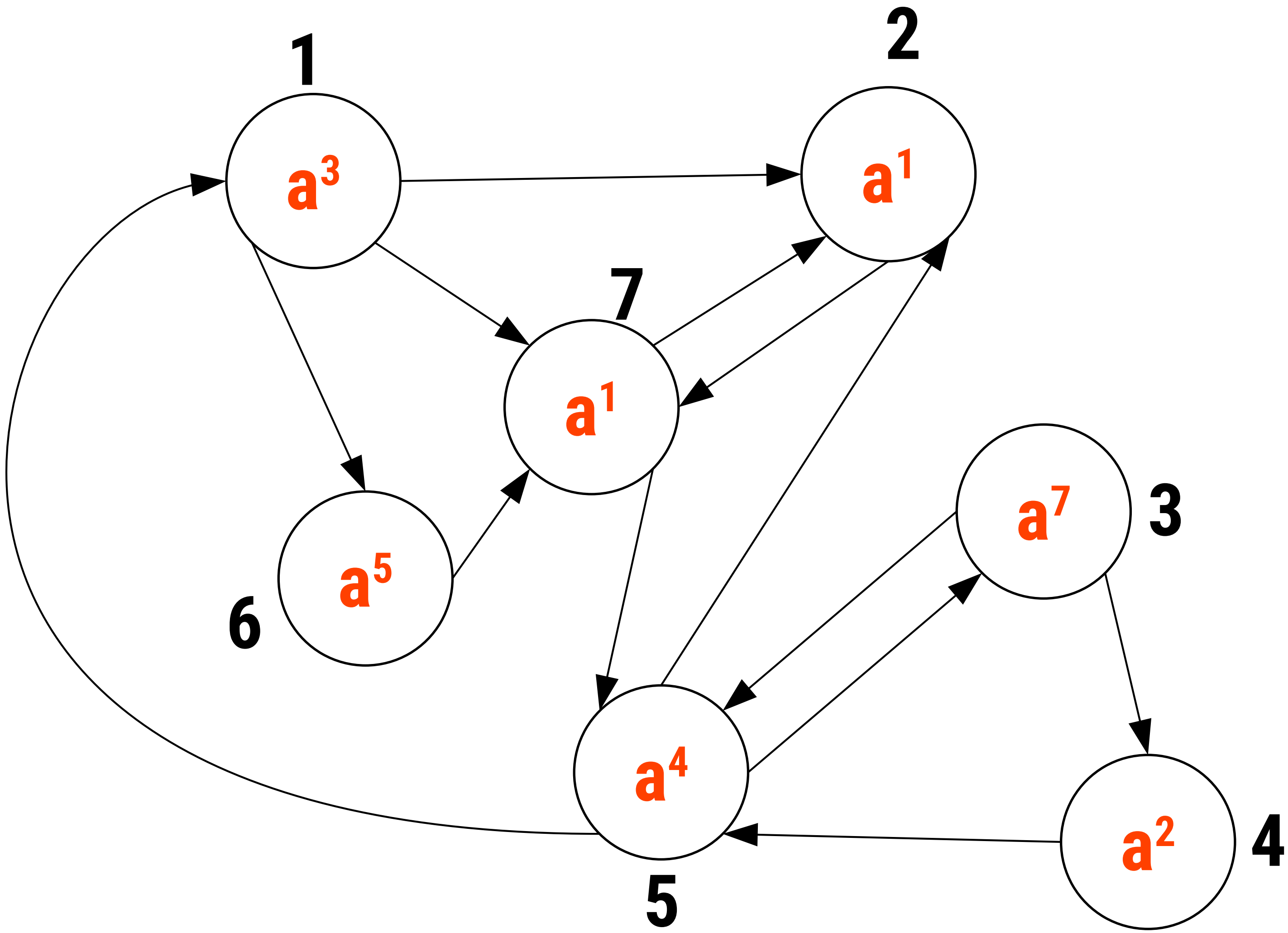
[4] *Martín-Videa*, C., *Păun*, G., *Pazos*, J., *Rodríguez-Patón*, A.. 2003. *Tissue P Systems*. In *Theoretical Computer Science.* Elsevier.

Spiking Neural P Systems

O = {a}

a - spike

t=0



Rule in Neuron 1: $a^3/a^2 \rightarrow a: 3$

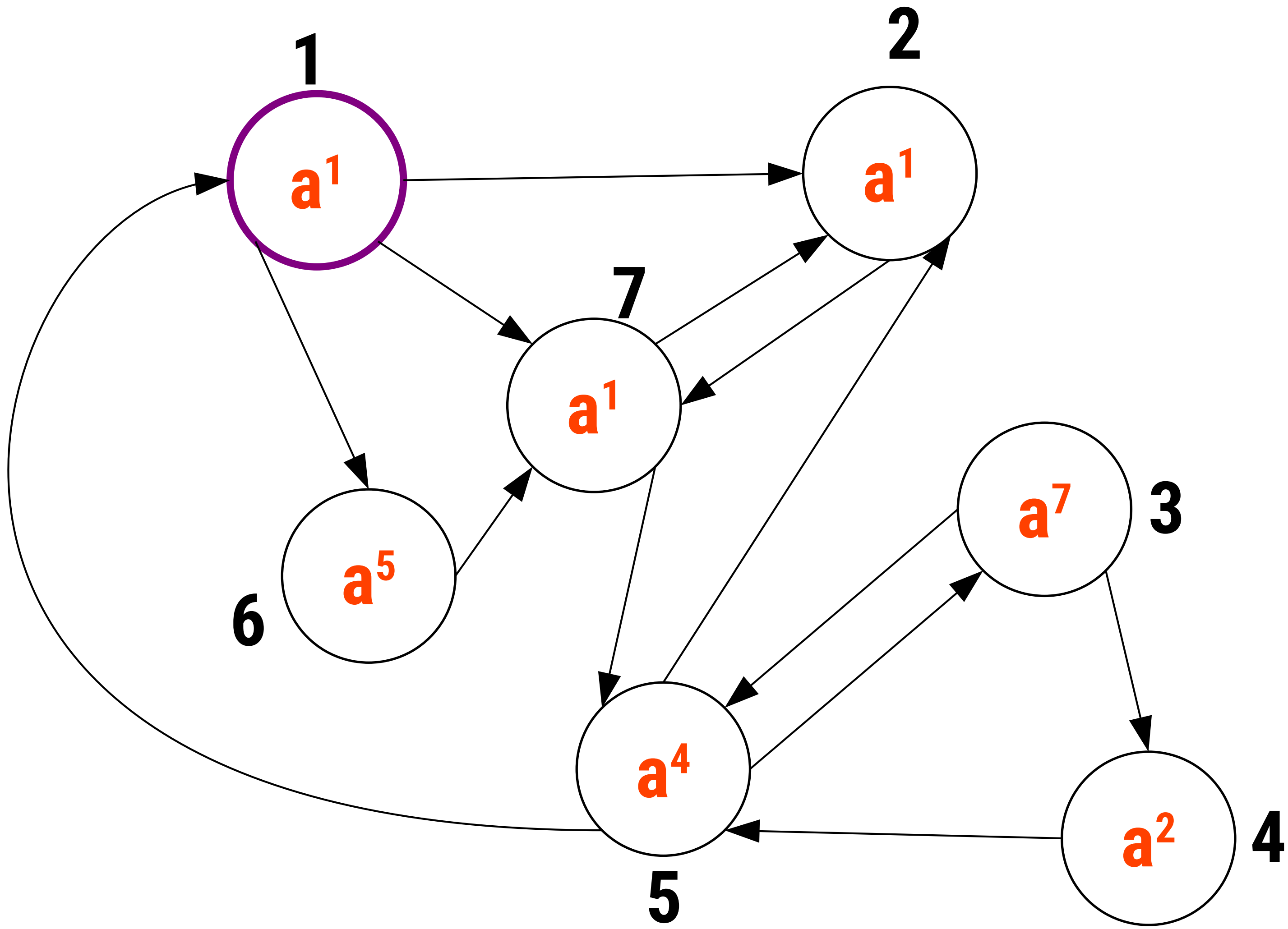
[5] *Ionescu*, M., Păun, G., *Yokomori*, T.. 2006. *Spiking Neural P Systems*. In *Fundamenta Informaticae*. vol 71, issue 2,3 (February 2006), 279-308.

Spiking Neural P Systems

O = {a}

a - spike

t=0



Rule in Neuron 1: $a^3/a^2 \rightarrow a: 3$

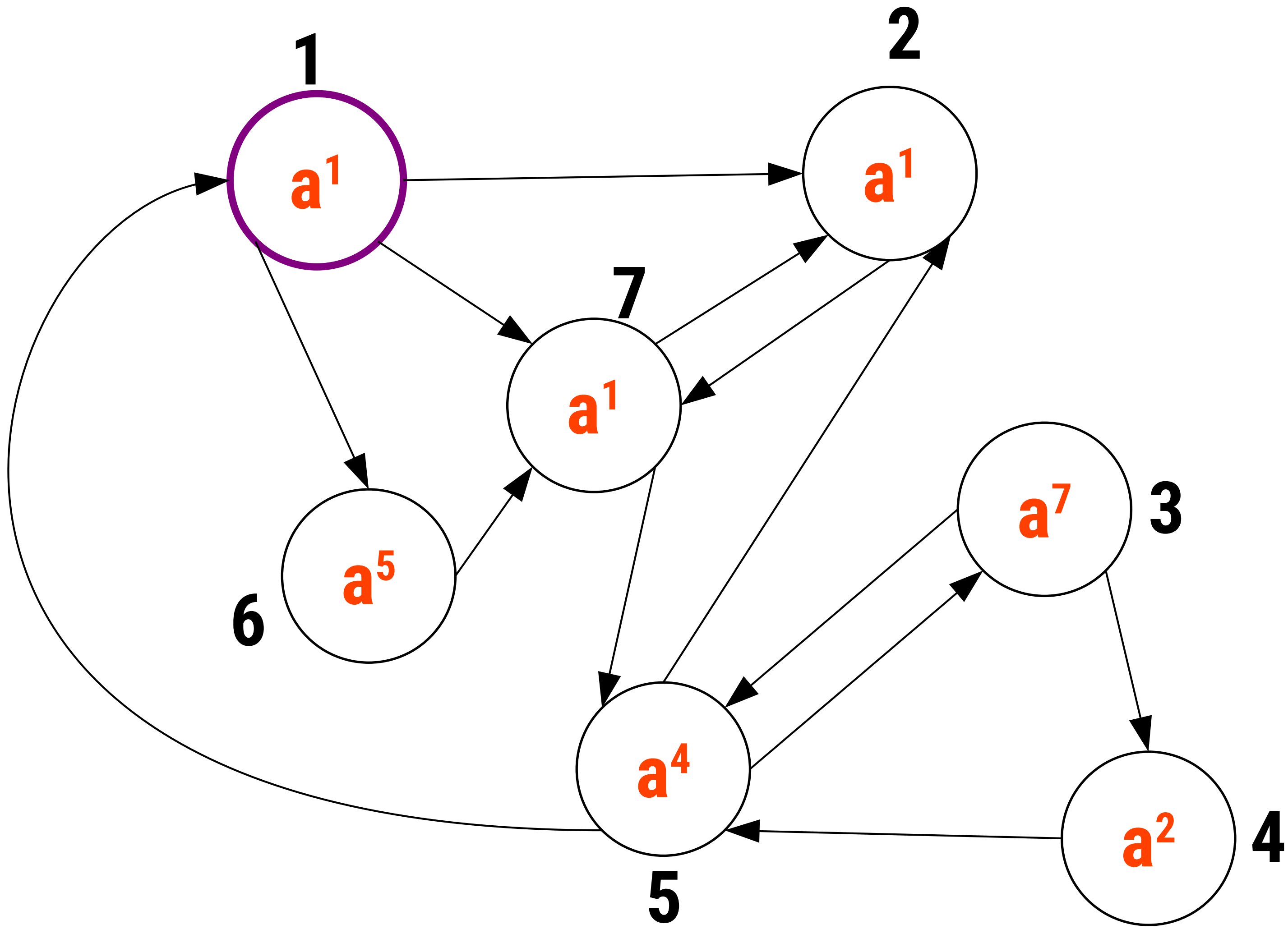
[5] *Ionescu*, M., Păun, G., *Yokomori*, T.. 2006. *Spiking Neural P Systems*. In *Fundamenta Informaticae*. vol 71, issue 2,3 (February 2006), 279-308.

Spiking Neural P Systems

O = {a}

a - spike

t=1



Rule in Neuron 1: $a^3/a^2 \rightarrow a: 3$

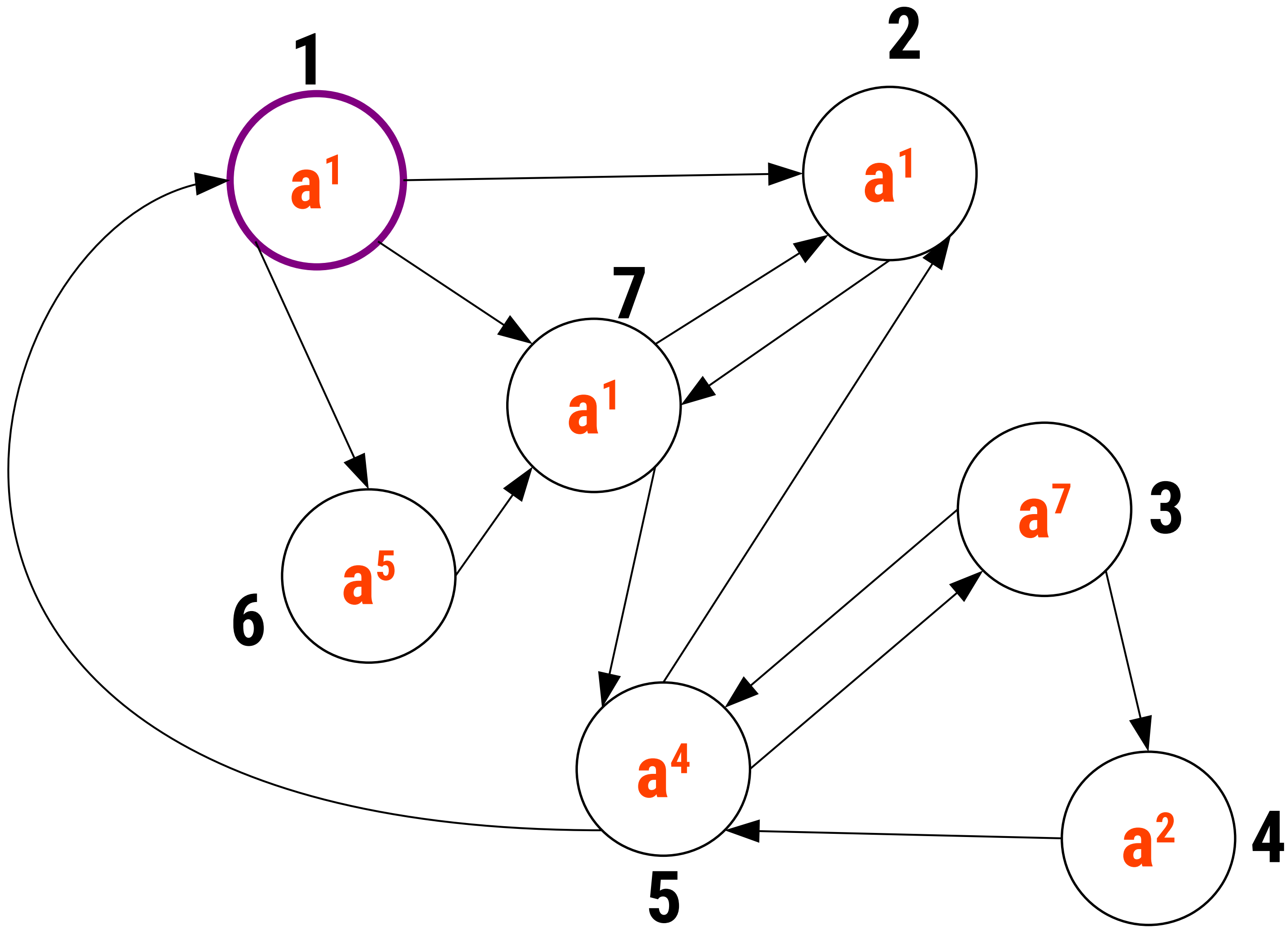
[5] *Ionescu*, M., Păun, G., *Yokomori*, T.. 2006. *Spiking Neural P Systems*. In *Fundamenta Informaticae*. vol 71, issue 2,3 (February 2006), 279-308.

Spiking Neural P Systems

O = {a}

a - spike

t=2



Rule in Neuron 1: $a^3/a^2 \rightarrow a: 3$

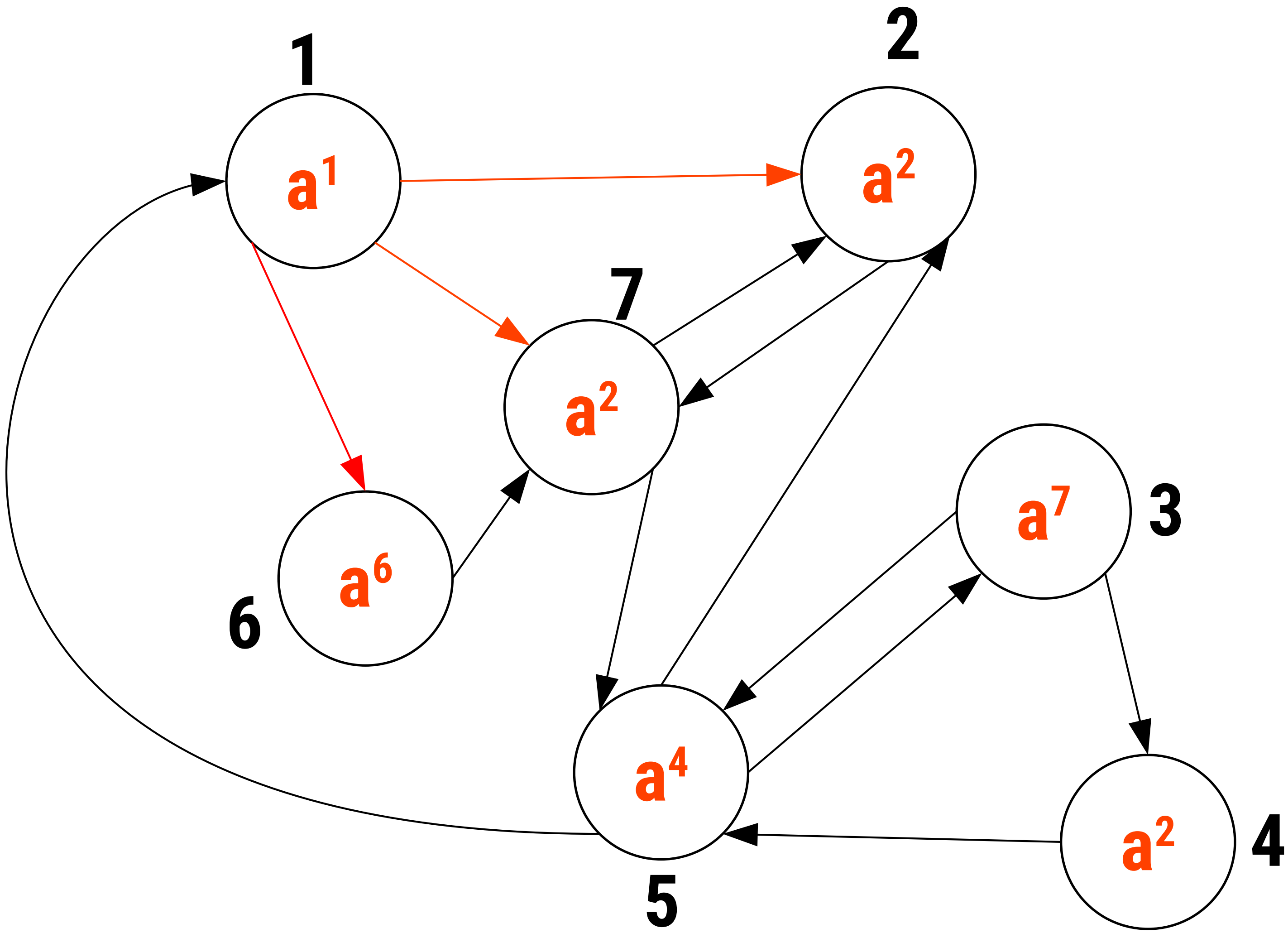
[5] *Ionescu*, M., Păun, G., *Yokomori*, T.. 2006. *Spiking Neural P Systems*. In *Fundamenta Informaticae*. vol 71, issue 2,3 (February 2006), 279-308.

Spiking Neural P Systems

O = {a}

a - spike

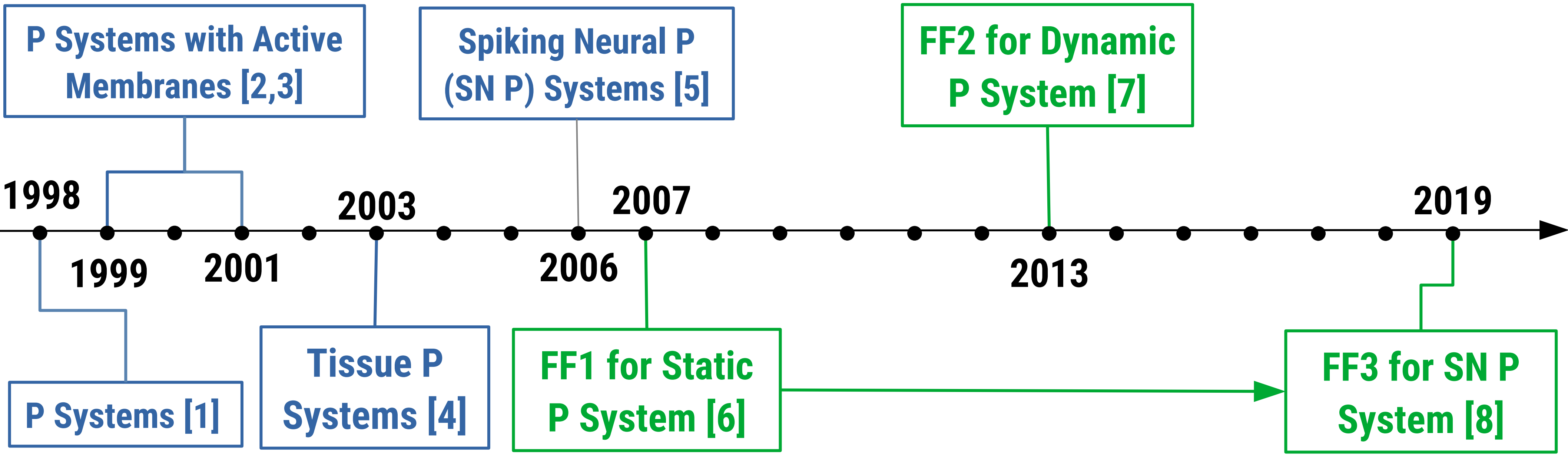
t=3



Rule in Neuron 1: $a^3/a^2 \rightarrow a: 3$

[5] *Ionescu*, M., Păun, G., *Yokomori*, T.. 2006. *Spiking Neural P Systems*. In *Fundamenta Informaticae*. vol 71, issue 2,3 (February 2006), 279-308.

P Systems and Formal Frameworks (FFs)



[1] *Păun*, G.. 1998. *Computing with Membranes*. In *Technical Report. Turku Centre for Computer Science*.

[2] *Păun*, G.. 1999. *P Systems with Active Membranes: Attacking NP-Complete Problems*. In *Centre for Discrete Mathematics and Theoretical Computer Science* (CDMTCS-102) – Research Report Series

[3] *Păun*, G.. 2001. *P Systems with Active Membranes: Attacking NP-Complete Problems*. In *Journal of Automata, Languags, and Combinatorics.* vol.6, issue 1 (January 2001), 75-90.

[4] *Martín-Videa*, C., *Păun*, G., *Pazos*, J., *Rodríguez-Patón*, A.. 2003. *Tissue P Systems*. In *Theoretical Computer Science.* Elsevier.

[5] *Ionescu*, M., Păun, G., *Yokomori*, T.. 2006. *Spiking Neural P Systems*. In *Fundamenta Informaticae*. vol 71, issue 2,3 (February 2006), 279-308.

[6] *Freund* R., *Verlan* S. 2007. *A Formal Framework for Static (Tissue) P Systems*. In: Eleftherakis G., Kefalas P., Păun G., Rozenberg G., Salomaa A. (eds) *Membrane Computing*. WMC 2007. Lecture Notes in Computer Science, vol 4860. Springer, Berlin, Heidelberg

[7] *Freund*, R., *Pérez-Hurtado*, I., *Riscos-Núñez*, A., *Verlan*, S.. 2013. *A Formalization of Membrane Systems with Dynamically Evolving Structures.* In *International Journal of Computer Mathematics*, 90:4, 801-815

[8] *Verlan*, S., *Freund*, R., *Alhazov*, A., *Pan*, L.. 2008. A Formal Framework for Spiking Neural P Systems. In *Proceedings of 20th Conference on Membrane Computing (CMC20)*

Formal Framework 1 – Network of Cells

Network of Cells: $\Pi = (n, V, w, Inf, R)$

n – Number of Cells

V – Alphabet of Objects

$w = (w_1,\ldots,w_i,\ldots,w_n)$ – Vector of Multisets

w_i – Multiset in Cell **i**

$Inf = (Inf_1,\ldots,Inf_i,\ldots,Inf_n)$ – Vector of Sets

Inf_i – Set of objects ocurring infinitely often in Cell **i**

R – Set of Rules

Formal Framework 1 – Network of Cells

R

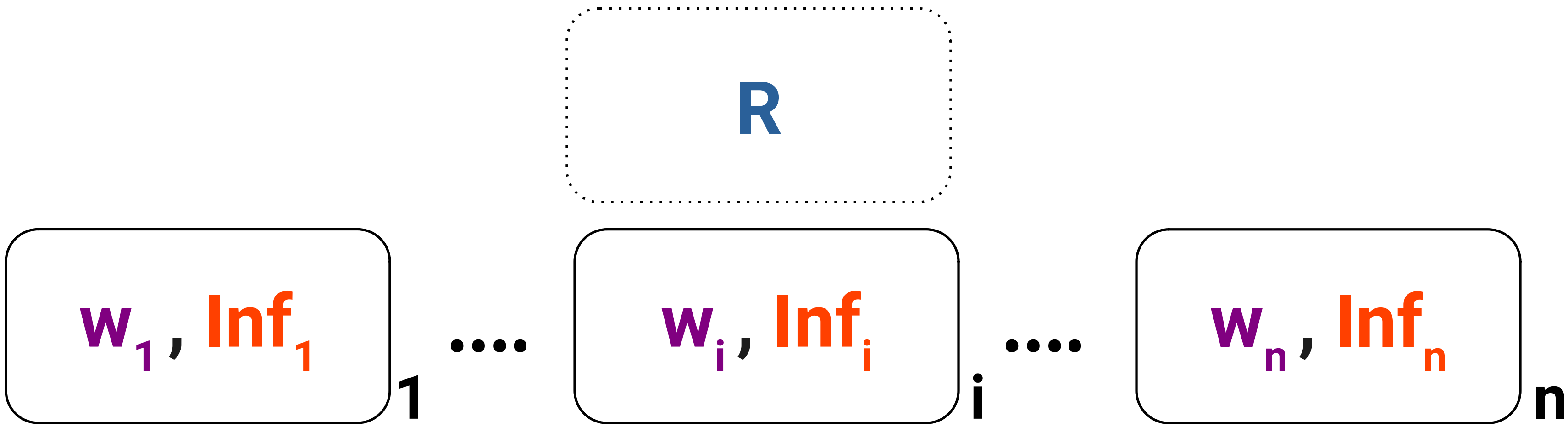
w_1, Inf_1 $_1$ …. w_i, Inf_i $_i$ …. w_n, Inf_n $_n$

$\Pi = (n=3, v=\{a,b,c,d\}, w=\{ab,bc,cc\}, Inf=(\{d\},\{\},\{a,b\}), R)$

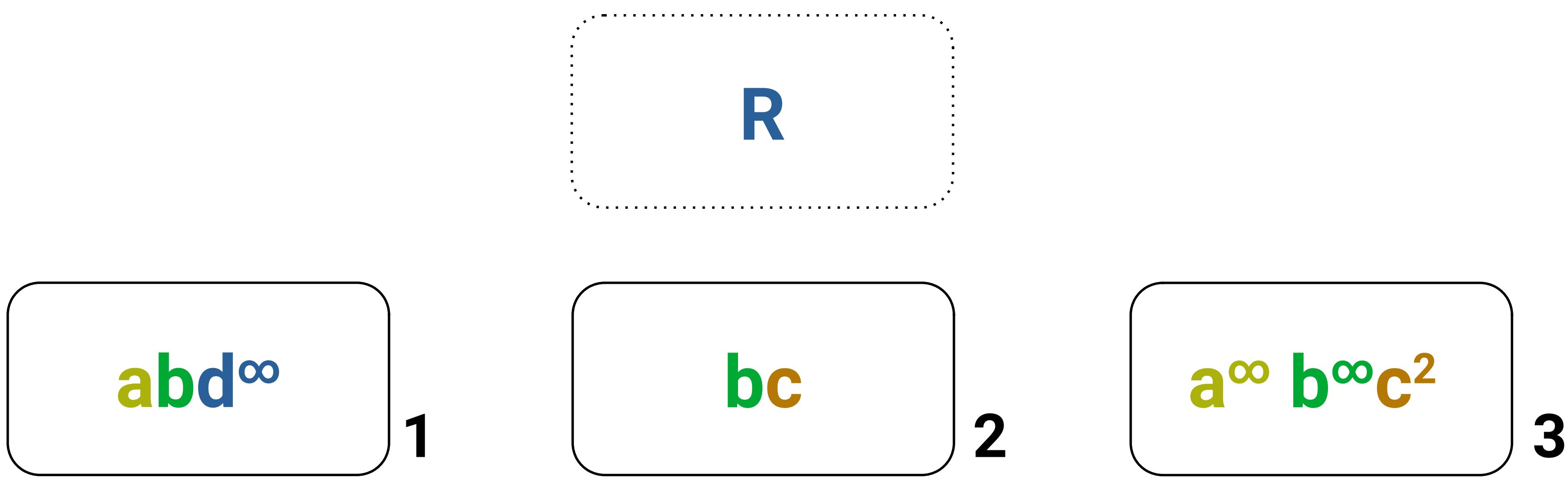
R

ab, d^∞ $_1$ bc $_2$ $cc, a^\infty b^\infty$ $_3$

Formal Framework 1 – Network of Cells



$$\Pi = (n=3, v=\{a,b,c,d\}, w=\{ab,bc,cc\}, Inf=(\{d\},\{\},\{a,b\}), R)$$



Formal Framework 1 – Interactive Rule

$$R = \{r\}$$

$r = X \rightarrow Y; P, Q;$

$X=(x_1,\ldots,x_n)$ – vector of multisets to be consumed

$Y=(y_1,\ldots,y_n)$ – vector of multisets to be produced.

$P = (p_1,\ldots,p_n)$ – vector of required multisets.

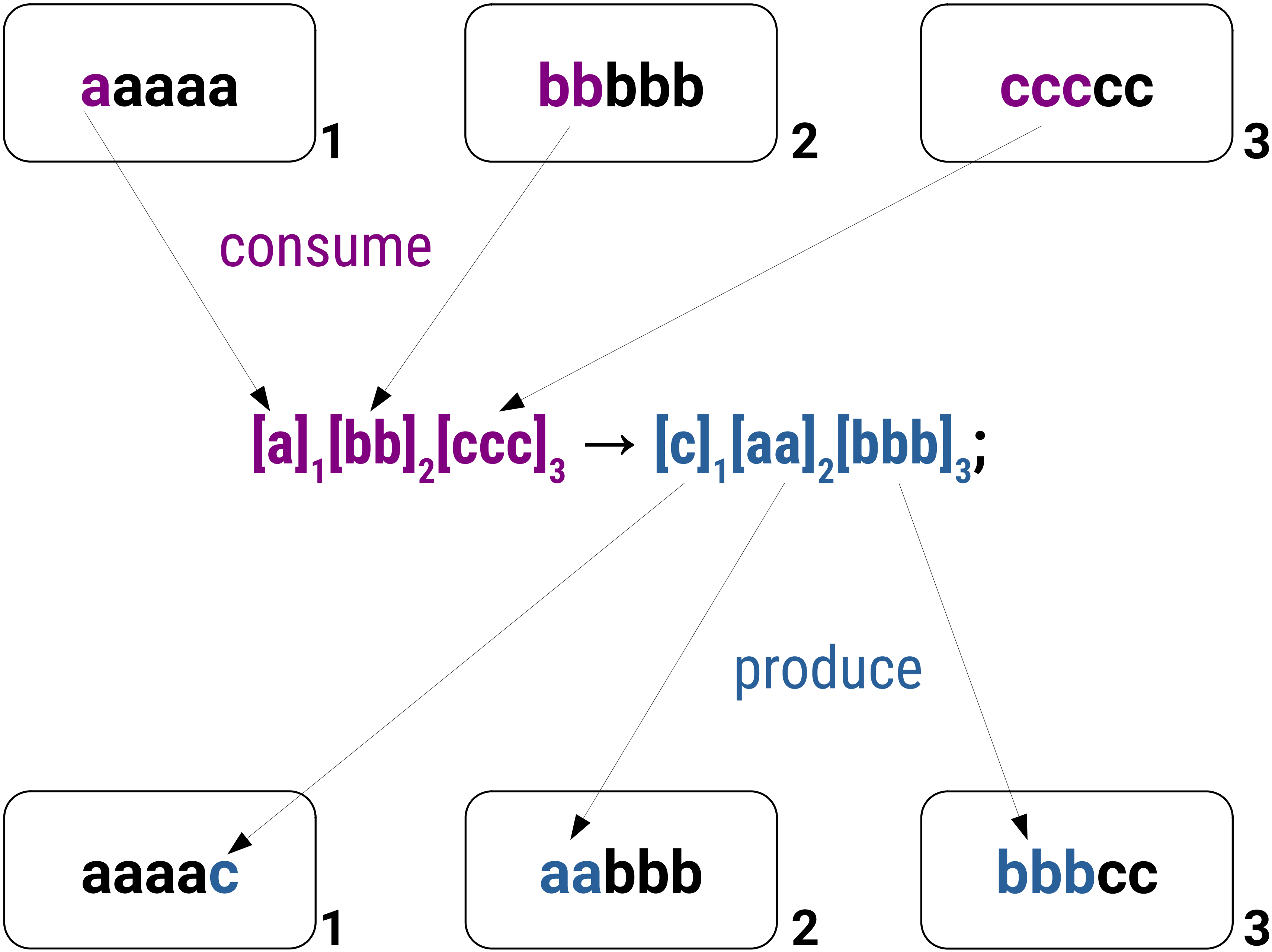
$Q = (q_1,\ldots,q_n)$ – vector of forbidden multisets.

$$\boxed{x_1, y_1, p_1, q_1}_1 \;\ldots\; \boxed{x_i, y_i, p_i, q_i}_i \;\ldots\; \boxed{x_n, y_n, p_n, q_n}_n$$

$$[x_1]_1\ldots[x_n]_n \rightarrow [y_1]_1\ldots[y_n]_n;\ [p_1]_1\ldots[p_n]_n,\ [q_1]_1\ldots[q_n]_n;$$

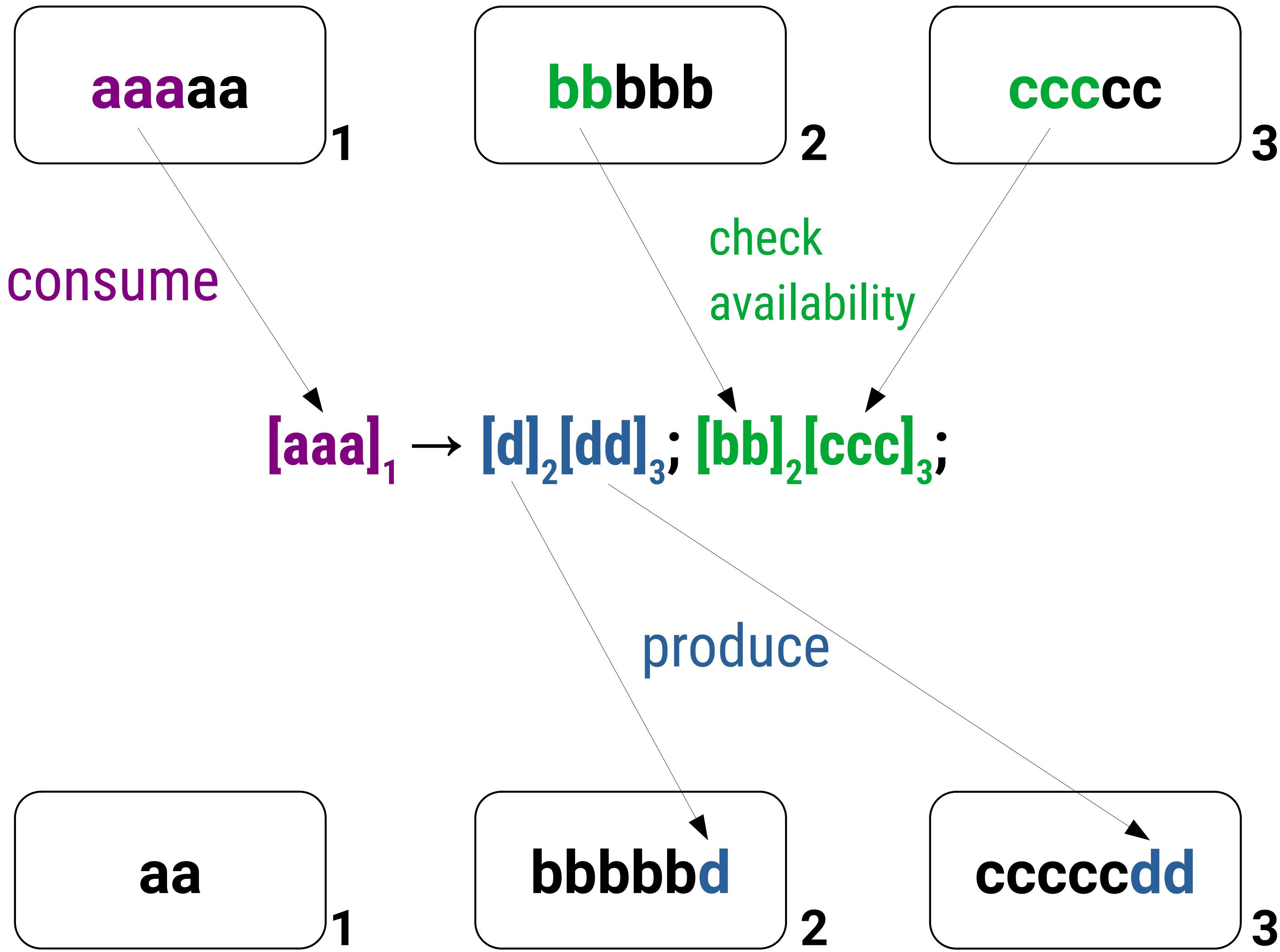
Formal Framework 1 – Interactive Rule

$\Pi = (n=3, v=\{a,b,c,d\}, w=\{\ldots\}, Inf=(\{\},\{\},\{\}), R)$



Formal Framework 1 – Interactive Rule

$\Pi = (n=3, v=\{a,b,c,d\}, w=\{\ldots\}, Inf=(\{\},\{\},\{\}), R)$



Formal Framework 1 – Interactive Rule

$\Pi = (n=3, v=\{a,b,c,d\}, w=\{\ldots\}, Inf=(\{\},\{\},\{\}), R)$

aaaaa	bbbbb	ccccc
1	2	3

to consume

check availability

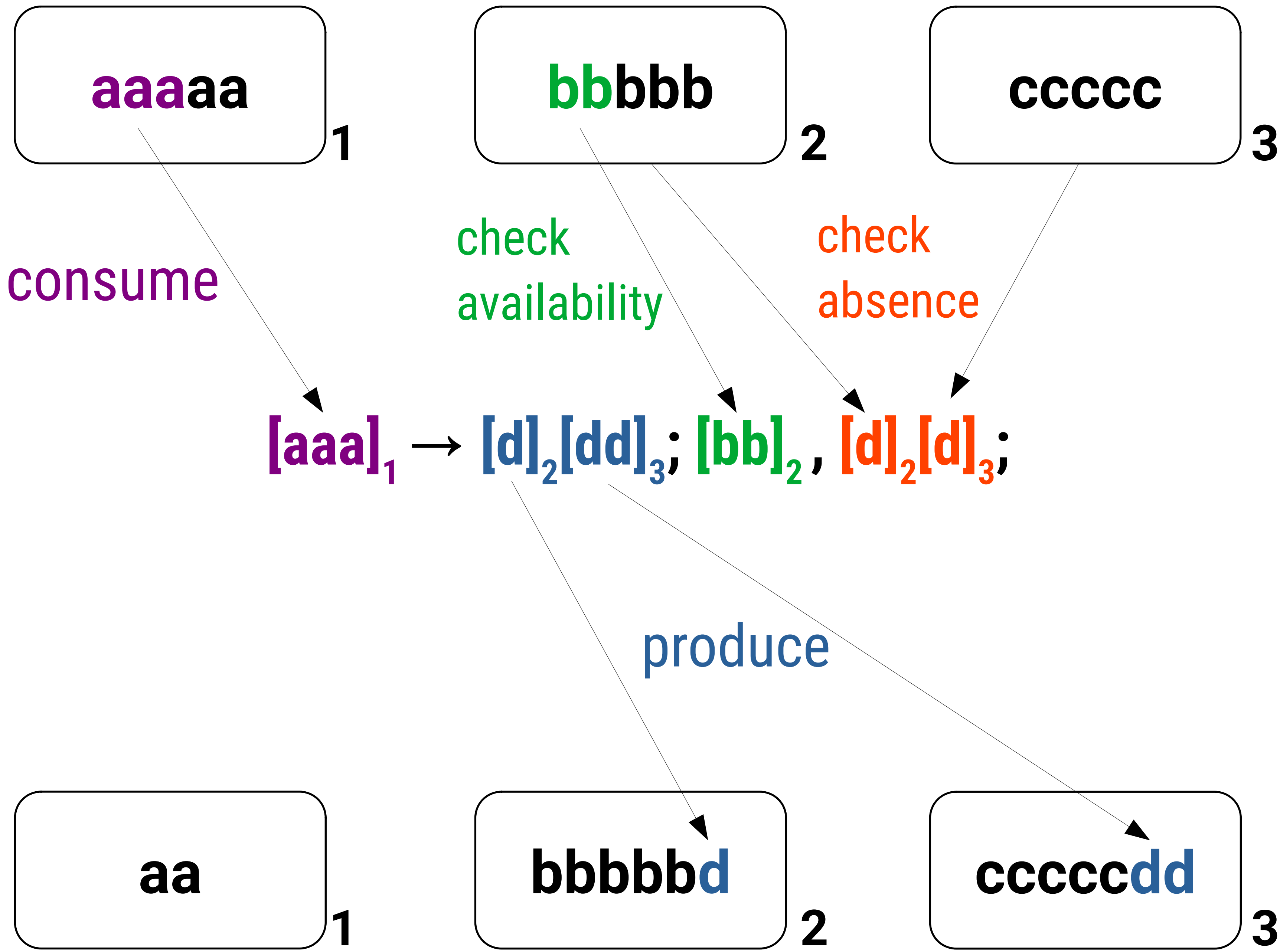
$[aaa]_1 \rightarrow [d]_2[dd]_3; [bbd]_2[ccc]_3;$

The rule is NOT ***eligible*** since multiset **bbd** is required to be in cell 2.

aaaaa	bbbbb	ccccc
1	2	3

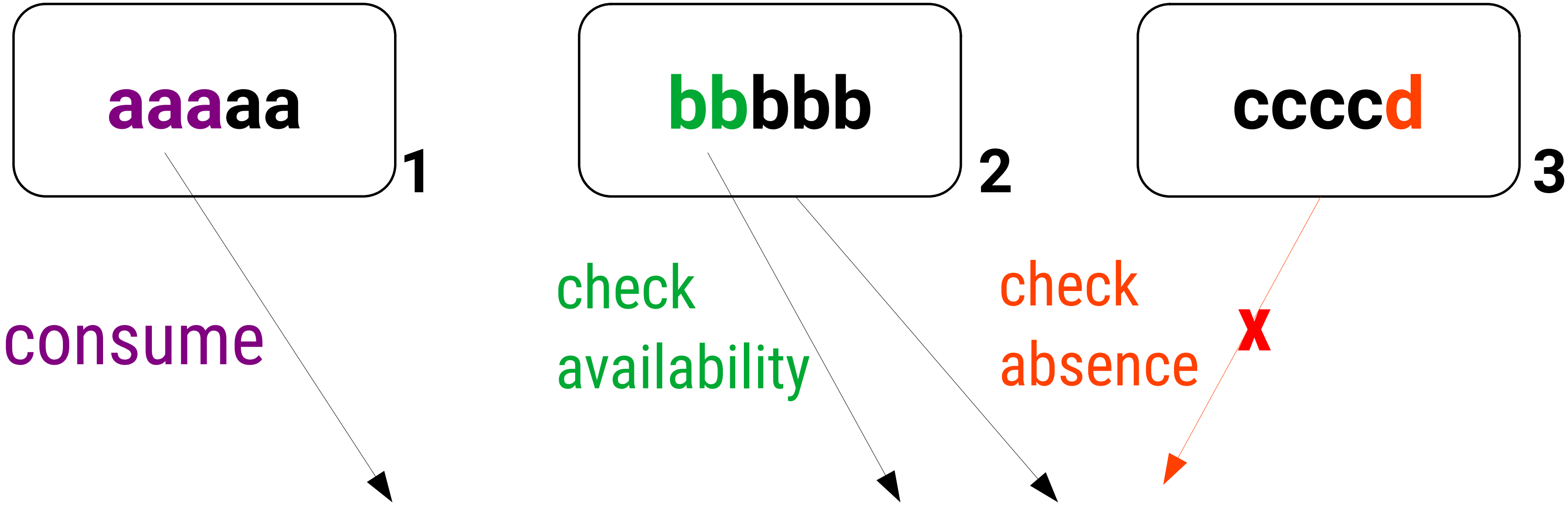
Formal Framework 1 – Interactive Rule

$$\Pi = (n=3,\ v=\{a,b,c,d\},\ w=\{\ldots\},\ Inf=(\{\},\{\},\{\}),\ R)$$



Formal Framework 1 – Interactive Rule

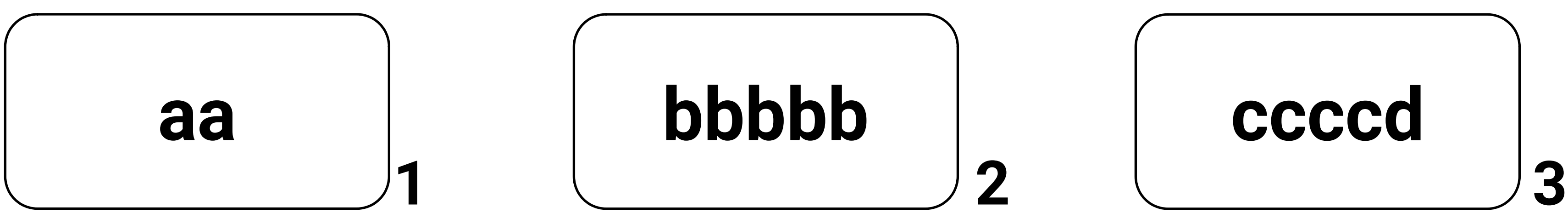
$$\Pi = (n=3,\ v=\{a,b,c,d\},\ w=\{\ldots\},\ Inf=(\{\},\{\},\{\}),\ R)$$



$$[aaa]_1 \rightarrow [d]_2[dd]_3;\ [bb]_2,\ [d]_2[d]_3;$$

The rule is NOT ***eligible*** since multiset

d is forbidden in cell 3.



Formal Framework 1 – Rule Eligibility

$r = X \rightarrow Y; P, Q;$

$X=(x_1,\ldots,x_n)$ – vector of multisets to be consumed

$Y=(y_1,\ldots,y_n)$ – vector of multisets to be produced.

$P = (p_1,\ldots,p_n)$ – vector of required multisets.

$Q = (q_1,\ldots,q_n)$ – vector of forbidden multisets.

$$C^f = \boxed{w_1}_1 \;\ldots\; \boxed{w_i}_i \;\ldots\; \boxed{w_n}_n$$

$$r = \boxed{x_1, y_1, p_1, q_1}_1 \;\ldots\; \boxed{x_i, y_i, p_i, q_i}_i \;\ldots\; \boxed{x_n, y_n, p_n, q_n}_n$$

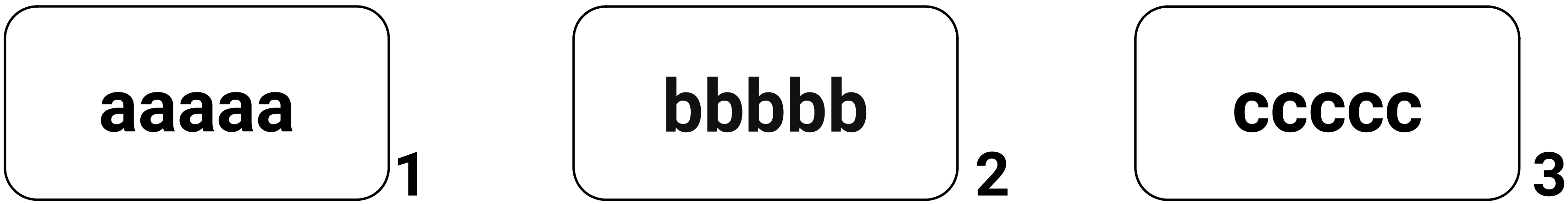
Conditions for **Rule Eligibility:**

$x_i \subseteq w_i$

$p_i \subseteq w_i$

$q_i \not\subseteq w_i$

Formal Framework 1 – Applicable Multiset of Rules



Eligible Rules:

r_1: $[aa]_1 \rightarrow [dd]_2$;

r_2: $[b]_2 \rightarrow [d]_1[d]_3$;

r_3: $[aaa]_1[cccc]_3 \rightarrow [bb]_2$;

Applicable Multiset of Rules:

$R_1' = \{ r_1, r_1 \}$ $\quad R_5' = \{ r_3 \}$

$R_2' = \{ r_1, r_2 \}$ $\quad R_6' = \{ r_1 \}$

$R_3' = \{ r_2, r_2, r_2, r_2, r_2 \}$

$R_4' = \{ r_1, r_1, r_2, r_2, r_2, r_2, r_2 \}$

$R_7' = \{ r_2 \}$

Non-applicable Multiset of Rules:

$R_8' = \{ r_3, r_3 \}$ $\quad R_9' = \{ r_1, r_1, r_3 \}$ $\quad R_{10}' = \{ r_2, r_2, r_2, r_2, r_2, r_2 \}$

Formal Framework 1 – Derivation Modes

aaaaa $_1$	bbbbb $_2$	ccccc $_3$

Eligible Rules:

r_1: $[aa]_1 \rightarrow [dd]_2$;

r_2: $[b]_2 \rightarrow [d]_1[d]_3$;

r_3: $[aaa]_1[cccc]_3 \rightarrow [bb]_2$;

Applicable Multisets of Rules:

$R_1' = \{ r_1, r_1 \}$ $\quad$ $R_5' = \{ r_3 \}$

$R_2' = \{ r_1, r_2 \}$ $\quad$ $R_6' = \{ r_1 \}$

$R_3' = \{ r_2, r_2, r_2, r_2, r_2 \}$

$R_4' = \{ r_1, r_1, r_2, r_2, r_2, r_2, r_2 \}$

$R_7' = \{ r_2 \}$

Which of the applicable multisets or rules should be used?

Formal Framework 1 – Derivation Mode - Sequential

$[\,aaaaa\,]_1 \quad [\,bbbbb\,]_2 \quad [\,ccccc\,]_3$

Eligible Rules:

r_1: $[aa]_1 \rightarrow [dd]_2$;

r_2: $[b]_2 \rightarrow [d]_1[d]_3$;

r_3: $[aaa]_1[cccc]_3 \rightarrow [bb]_2$;

Applicable Multisets of Rules:

$R_1' = \{ r_1, r_1 \}$

$R_2' = \{ r_1, r_2 \}$

$R_3' = \{ r_2, r_2, r_2, r_2, r_2 \}$

$R_4' = \{ r_1, r_1, r_2, r_2, r_2, r_2, r_2 \}$

$R_5' = \{ r_3 \}$

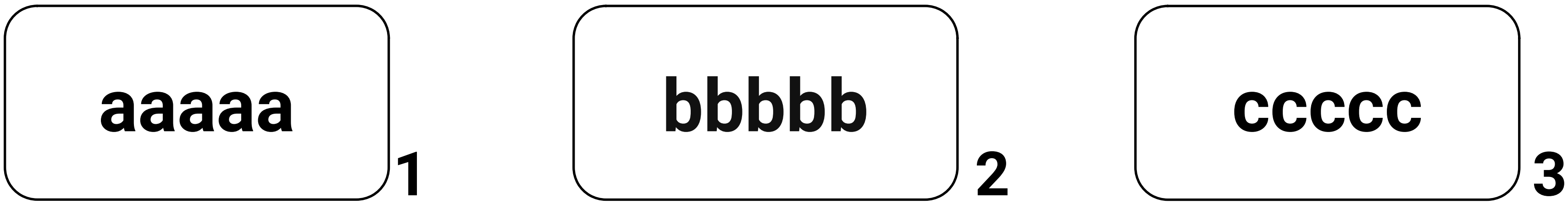
$R_6' = \{ r_1 \}$

$R_7' = \{ r_2 \}$

Usable Multisets in Sequential Mode:

$R_6' = \{ r_1 \}$ $\quad R_7' = \{ r_2 \}$ $\quad R_5' = \{ r_3 \}$

Formal Framework 1 – Derivation Mode – Asynchronous



Eligible Rules:

r_1: $[aa]_1 \rightarrow [dd]_2$;

r_2: $[b]_2 \rightarrow [d]_1[d]_3$;

r_3: $[aaa]_1[cccc]_3 \rightarrow [bb]_2$;

Applicable Multisets of Rules:

$R_1' = \{ r_1, r_1 \}$

$R_2' = \{ r_1, r_2 \}$

$R_3' = \{ r_2, r_2, r_2, r_2, r_2 \}$

$R_4' = \{ r_1, r_1, r_2, r_2, r_2, r_2, r_2 \}$

$R_5' = \{ r_3 \}$

$R_6' = \{ r_1 \}$

$R_7' = \{ r_2 \}$

Usable Multisets in Asynchronous Mode:

Any applicable multiset R’

Formal Framework 1 – Derivation Mode – Maximally Parallel

$\boxed{aaaaa}_1 \quad \boxed{bbbbb}_2 \quad \boxed{ccccc}_3$

Eligible Rules:

r_1: $[aa]_1 \rightarrow [dd]_2$;

r_2: $[b]_2 \rightarrow [d]_1[d]_3$;

r_3: $[aaa]_1[cccc]_3 \rightarrow [bb]_2$;

Applicable Multisets of Rules:

$R_1' = \{ r_1, r_1 \}$

$R_2' = \{ r_1, r_2 \}$

$R_3' = \{ r_2, r_2, r_2, r_2, r_2 \}$

$R_4' = \{ r_1, r_1, r_2, r_2, r_2, r_2, r_2 \}$

$R_5' = \{ r_3 \}$

$R_6' = \{ r_1 \}$

$R_7' = \{ r_2 \}$

Usable Multisets in Maximally Parallel Mode:

$R_4' = \{ r_1, r_1, r_2, r_2, r_2, r_2, r_2 \}$ $R_8' = \{ r_1, r_3, r_2, r_2, r_2, r_2, r_2 \}$

Formal Framework 2 – Derivation Mode

2.6 Derivation Modes

1. $Appl(\Pi, C, \delta) \subseteq Appl(\Pi, C)$ - Set of applicable multisets of rules in δ-mode.

2. $Appl(\Pi, C, asyn) = Appl(\Pi, C)$ - *Asynchronous* Mode ($\delta = asyn$).

3. $Appl(\Pi, C, sequ) = \{R' \in Appl(\Pi, C) \mid |R'| = 1\}$ - *Sequential* Mode ($\delta = sequ$) - One rule per step.

4. $Appl(\Pi, C, max) = \{R' \in Appl(\Pi, C) \mid \not\exists R'' \in Appl(\Pi, C),\ R'i \not\subseteq R''\}$ - *Maximally Parallel* Mode ($\delta = max$) - Adding any rule to a maximally parallel R' will result in an inapplicable multiset of rules.

5. $Appl(\Pi, C, min) = \{R' \in Appl(\Pi, C) \mid \not\exists R'' \in Appl(\Pi, C),\ R' \subseteq R'', \exists j, (R'' - R') \cap R_j \neq \emptyset, R' \cap R_j = \emptyset\}$ - *Minimally Parallel* Mode ($\delta = min$) - There is no partition $R = R_1 \cup R_2 \cup \cdots \cup R_h$. Rule set R is be partitioned. $R' \subseteq R''$. R'' 'extends' R'.

6. $\delta \in \{asyn, sequ, max, min\}$ - Basic derivation modes

7. $Appl(\Pi, C, max_{rule}\delta) = \{R' \in Appl(\Pi, C, \delta) \mid \not\exists R'' \in Appl(\Pi, C, \delta)\ |R''| > |R'|\}$ - *Maximum Rules* δ-Mode.

8. $Appl(\Pi, C, max_{set}\delta) = \{R' \in Appl(\Pi, C, \delta) \mid \not\exists R'' \in Appl(\Pi, C, \delta)\ ||R''|| > ||R'||\}$ - *Maximum Sets (Partitions)* δ-Mode.

9. $Appl(\Pi, C, all_{set}\delta) = \{R' \in Appl(\Pi, C, \delta) \mid \forall j, 1 \leq j \leq h, (R_j \cap \bigcup_{X \in Appl(\Pi, C)} X \neq \emptyset) \to (R_j \cap R' \neq \emptyset)\}$ - *All Set* δ-Mode.

Formal Framework 2 – Transition & Halting Conditions

2.7 Transition

1. $[C \Rightarrow_{(\Pi,\Delta)} C'] \Leftrightarrow [[\exists R' \in Appl(\Pi, C, \Delta)][C' = Apply(\Pi, C, R')]]$ - *Transition* in Δ-mode.
2. $[C \Rightarrow^{*}_{(\Pi,\Delta)} C']$ - Transitive closure and reflexive nature of the transition relation $\Rightarrow_{(\Pi,\Delta)}$
3. $[accessible(C, \Pi, \Delta)] \Leftrightarrow [C_0 \Rightarrow_{(\Pi,\Delta)} C]$ - Accessibility of configuration C in Δ-mode.
4. $Acc(\Pi, \Delta) = \{C \mid accessible(C, \Pi, \Delta)\}$ - Set of all accessible configurations in Δ-mode derivation.
5. $[Deterministic(\Pi, \Delta)] \Leftrightarrow [\forall C \in Acc(\Pi, \Delta)][Appl(\Pi, C, \Delta)| \leq 1]$ - Determinism of system Π under Δ-mode derivation.

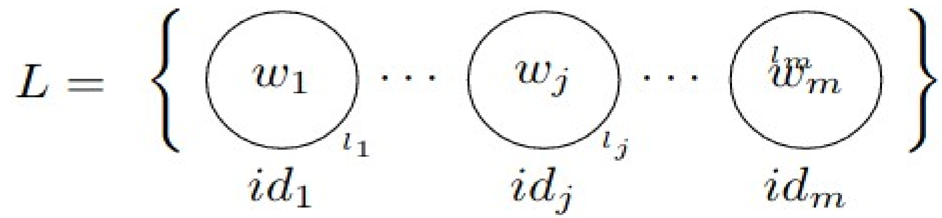
2.8 Halting Conditions

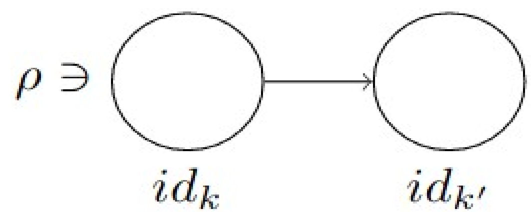
1. $H(\Pi, \Delta) = \{C' \in Acc(\Pi, \Delta) \mid Appl(\Pi, C', \Delta) = \emptyset\}$
   - Set of *total halting* configurations.
   - Accessible configurations where there are not applicable multisets of rules.
2. $A(\Pi, \Delta) = \{C' \in Acc(\Pi, \Delta) \mid Appl(\Pi, C', \Delta) \neq \emptyset, \forall R' \in Appl(\Pi, C', \Delta), Apply(\Pi, C', R') = C'\}$
   - Set of *adult halting* configurations.
   - Accessible configurations where for all applicable rule R' applying R' to configuration C' result to C'.
3. $h(\Pi, \Delta) = \{C' \in Acc(\Pi, \Delta) \mid \nexists R' \in Appl(\Pi, C', \Delta), \forall i, 1 \leq i \leq h, R' \cap R_j \neq \emptyset\}$
   - Set of *partial halting* configurations.
   - Accessible configurations there are no multiset R' of applicable rules such that R' contains rules from all partions R_j.

Preview of Formal Framework 2

Preview: Formal Framework 2 - Configuration

$$\mathcal{C} = (L, \rho) = (L = \{(id_1, l_1, w_1), ..., (id_j, l_j, w_j), ...(id_m, l_m, w_m)\}, \rho)$$





- $id_j \in \mathbb{N}$ - *id*
- $w_j \in O^\circ$ - *multiset* over O
- $l_j \in Lab$ - *label*
- (id_j, l_j, w_j) - *labelled cell*
- $L \in (\mathbb{N} \times Lab \times O^\circ)^*$ - list of labelled cells
- $\rho \subseteq \mathbb{N} \times \mathbb{N}$ - *relations* between cells (ids)
- $\mathcal{C} = (L, \rho)$ - *configuration*
- $\mathcal{C}_L = L$ and $\mathcal{C}_\rho = \rho$.

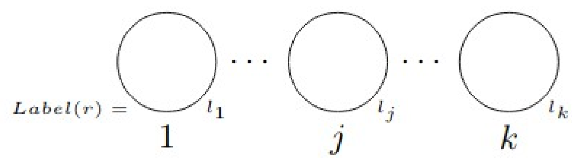
Preview: Formal Framework 2 - Rule

1.2 Components of a Rule

0. $r = (Labels, \rho, Perm, For, Rewrite, Label\text{-}Rename, Delete, Delete\text{-}and\text{-}Move, Generate, Generate\text{-}and\text{-}Copy, Change\text{-}Relation)$

    - r is a rule.

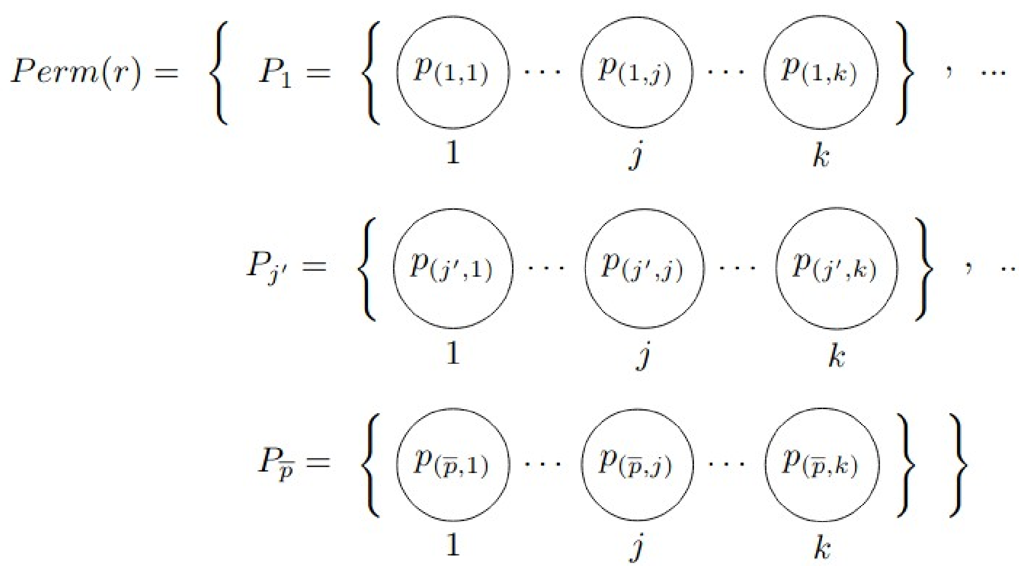
1. $Labels(r) = (l_1, ..., l_j...., l_k) \in Lab^k$



2. $\rho(r) \subseteq \mathbb{N}_k \times \mathbb{N}_k$

    - $\mathbb{N}_k = \{1, ..., j, ..., k\}$

3. $Perm(r) = \{P_1, ..., P_{j'}, ..., P_{\overline{p}}\} \subseteq \mathbb{C}_k$



Preview: Formal Framework 2 - Rule

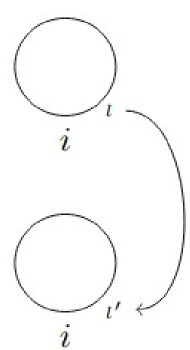
$$For(r) = \Bigg\{ \; F_1 = \Bigg\{ \underset{1}{f_{(1,1)}} \cdots \underset{j}{f_{(1,j)}} \cdots \underset{k}{f_{(1,k)}} \Bigg\}, \; \dots$$

$$F_{j'} = \Bigg\{ \underset{1}{f_{(j',1)}} \cdots \underset{j}{f_{(j',j)}} \cdots \underset{k}{f_{(j',k)}} \Bigg\}, \; \dots$$

$$F_{\overline{f}} = \Bigg\{ \underset{1}{f_{(\overline{f},1)}} \cdots \underset{j}{f_{(\overline{f},j)}} \cdots \underset{k}{f_{(\overline{f},k)}} \Bigg\} \; \Bigg\}$$

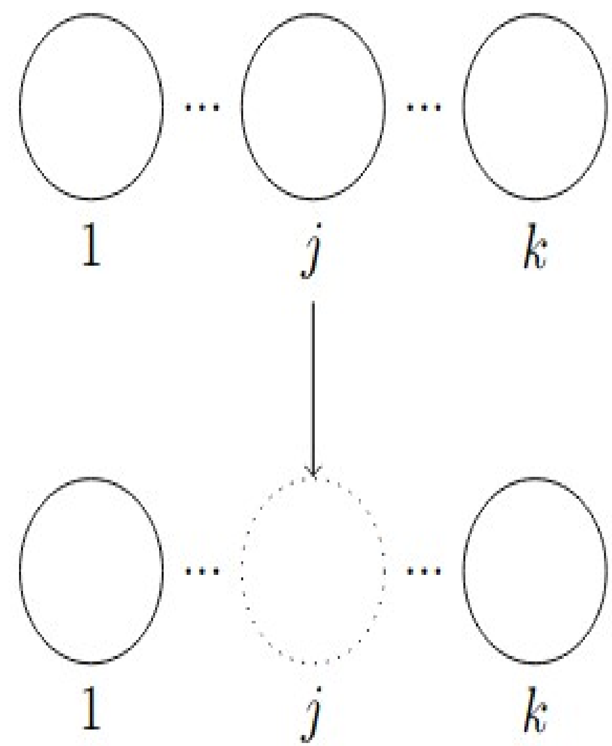
- $f_{(j',j)} \in O^{\circ}$

5. $Rewrite(r) = U \to V$
   - $U, V \in \mathbb{C}_k$
   - $Rewrite(r)$ is a general rewriting rule, rewriting a finite basic configuration U to another finite basic configuration V.
6. $Label\text{-}Rename(r) = \{..., (i, l'), ...\} \in (\mathbb{N}_k \times Lab)^*$



Preview: Formal Framework 2 - Rule

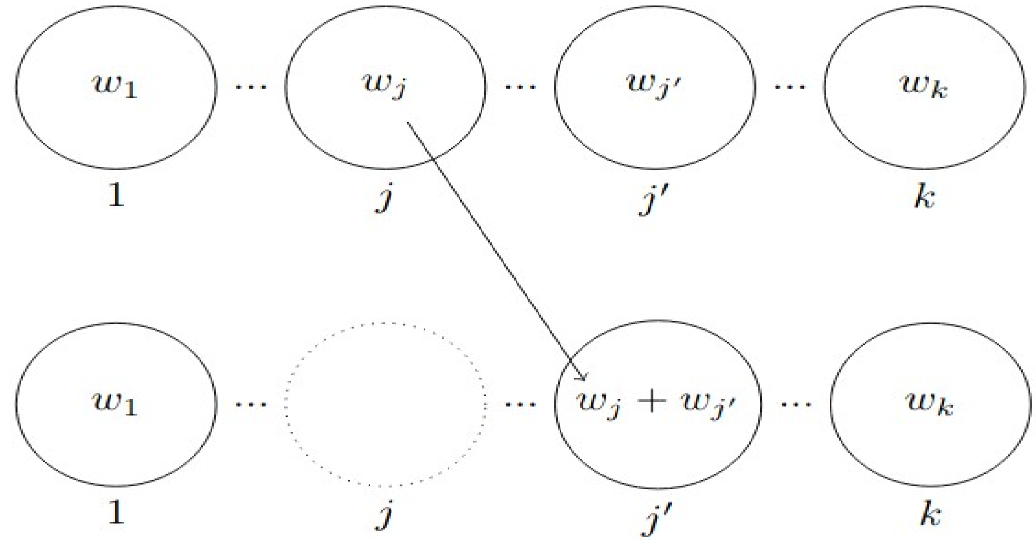
7. $Delete(r) = \{..., j, ...\} \in \mathbb{N}_k^*$



- j - id of cell to be deleted

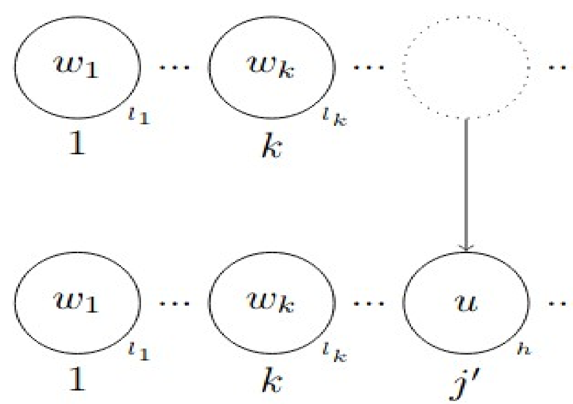
8. $Delete\text{-}Move(r) \in (\mathbb{N}_k \times \mathbb{N}_k)^*$

Preview: Formal Framework 2 - Rule



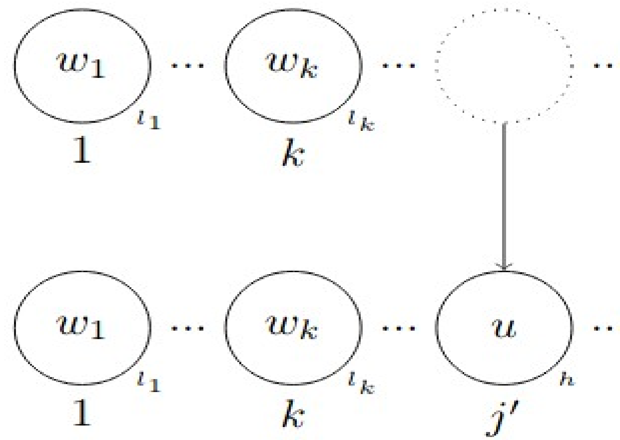
- (j, j') - *pair of ids*
- j - *id* of cell to be deleted
- j' - *id* of cell to receive the multiset
- *Delete-Move*(r)- *list of pairs* of ids

9. $Generate(r) = \{..., (j', h, u), ...\} \in (\mathbb{N}' \times Lab \times O^{\circ})^*$



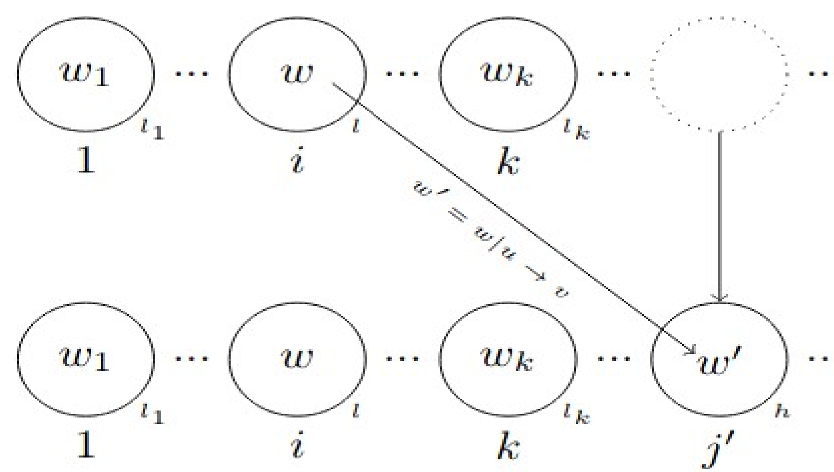
Preview: Formal Framework 2 - Rule

9. $Generate(r) = \{..., (j', h, u), ...\} \in (\mathbb{N}' \times Lab \times O^{\circ})^*$



- j' - *primed id* - new id
- h - label
- u - multiset

10. $Generate\text{-}Copy(r) = \{..., (j', h, i, (u, v)), ...\} \in (\mathbb{N}' \times Lab \times \mathbb{N} \times (O^{\circ} \times O^{\circ}))^*$



Preview of Formal Framework 3

Preview: Formal Framework 3 – Network of Cells

Definition 2. *A* network of cells of degree $n \geq 1$ *is a construct*

$$\Pi = (n, V, w, c_{in}, c_{out}, Inf, R)$$

where

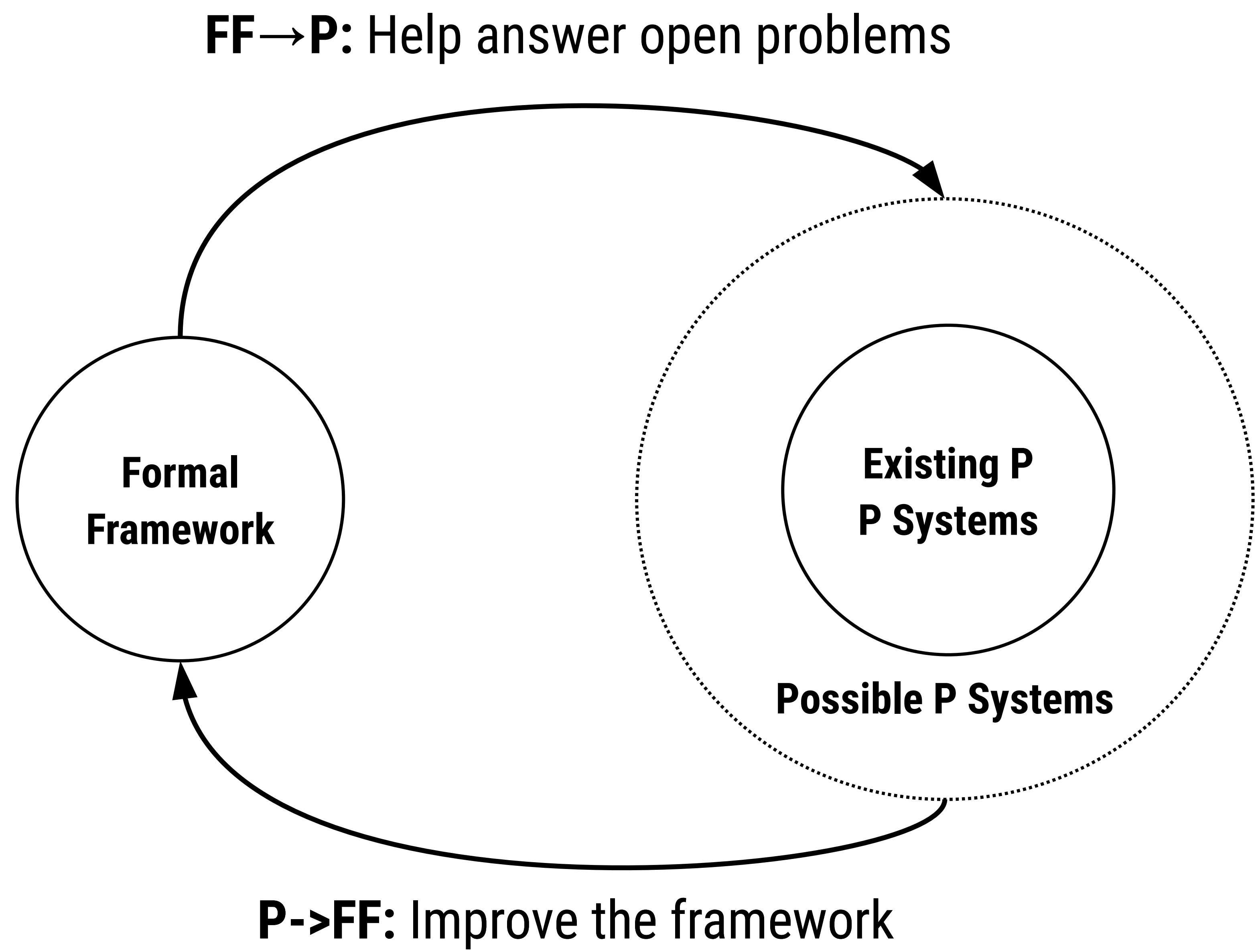
1. n *is the number of* cells*;*
2. V *is a finite alphabet;*
3. $w = (w_1, \ldots, w_n)$*,* $w_i \in \langle V, \mathbb{N} \rangle$*, for* $1 \leq i \leq n$*, is the* finite multiset initially associated to cell i*;*
4. $c_{in} \subseteq \{1, \ldots, n\}$ *is the set of* input *cells;*
5. $c_{out} \subseteq \{1, \ldots, n\}$ *is the set of* output *cells;*
6. $Inf = (Inf_1, \ldots, Inf_n)$*,* $Inf_i \subseteq V$*, for* $1 \leq i \leq n$*, is the* set of symbols occurring infinitely often in cell i *(in most of the cases, only one cell, called the* environment*, will contain symbols occurring with infinite multiplicity);*

Preview: Formal Framework 3 – Interactive Rules

Definition 4. *We say that an interaction rule* $r = (X \to Y; E)$ *is* eligible *for the configuration* C *with* $C = (u_1, \ldots, u_n)$ *if and only if for all* i, $1 \leq i \leq n$, *we have*

- $x_i \subseteq u_i$ *(x_i is a submultiset of u_i) and*
- $u_i \in L^\circ(E_i)$ *(u_i belongs to the regular multiset language described by the expression E_i).*

Formal Framework - Research Approaches



Formal Framework – Research Ideas

1. Merge the dynamic formal framework with the SNP formal framework . **(FF)**

2. *<u>Conjecture:</u>* Many SNP system variants are ‘equivalent’. Use formal framework to check if this is true. **(FF→ P).**

3. Extentend the formal framework to handle self-modifying P systems. **(P→FF).**

4. Check if “all” P systems if they can be represented using formal framework. Extend the framework if needed. **(P→ FF)**

5. Reformulate rule representation as bottom-up instead of top down. **(FF)**