Automata and Turing Complete Computation with Bio-Inspired Neural Networks

Jérémie Cabessa

LEMMA – Université Paris II

IBISC – Université d'Évry

May 7, 2019

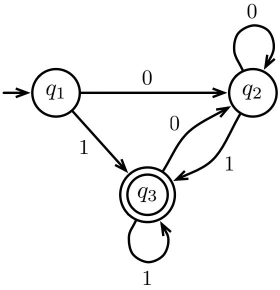
Introduction

- We recall important results about the computational capabilities of recurrent neural networks.
- We introduce a Turing complete bio-inspired paradigm for neural computation based on the concept of *synfire rings*.

Introduction

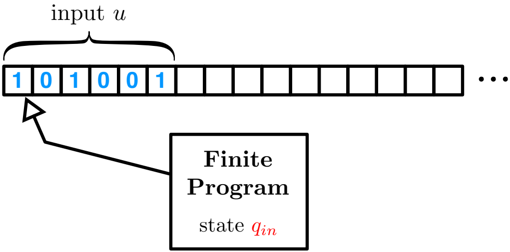
- We recall important results about the computational capabilities of recurrent neural networks.
- We introduce a Turing complete bio-inspired paradigm for neural computation based on the concept of *synfire rings*.

Finite State Automaton (FSA)



- input u is *accepted* by $\mathcal{A}$ if $\mathcal{A}(u)$ reaches a final state
- input u is *rejected* by $\mathcal{A}$ if $\mathcal{A}(u)$ otherwise

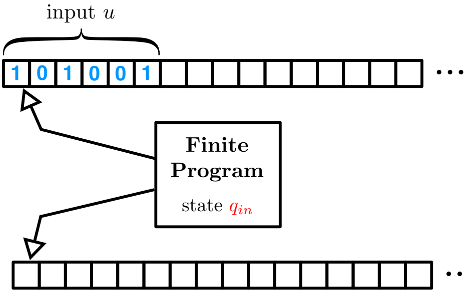
Turing Machine (TM)



- input u is *accepted* by $\mathcal{M}$ if $\mathcal{M}(u)$ reaches the state q_{acc}
- input u is *rejected* by $\mathcal{M}$ if $\mathcal{M}(u)$ reaches the state q_{rej}

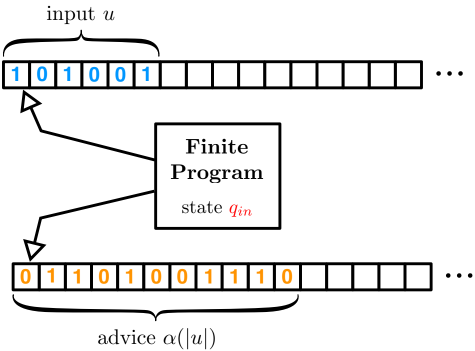
Turing Machine with Advice (TM/A)

Additional advice tape and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$



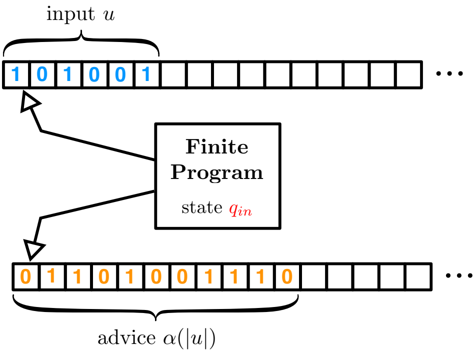
Turing Machine with Advice (TM/A)

Additional advice tape and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$



Turing Machine with Advice (TM/A)

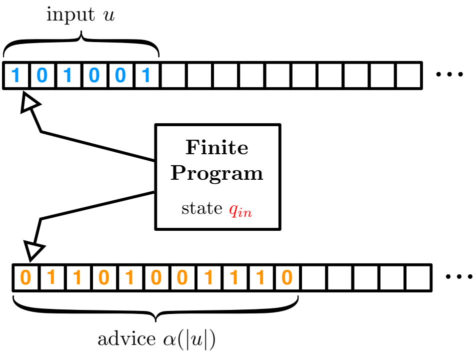
Additional advice tape and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$



- The class of languages recognized in polynomial time by Turing machines with polynomial advices (TM/poly(A)) is **P/poly**.

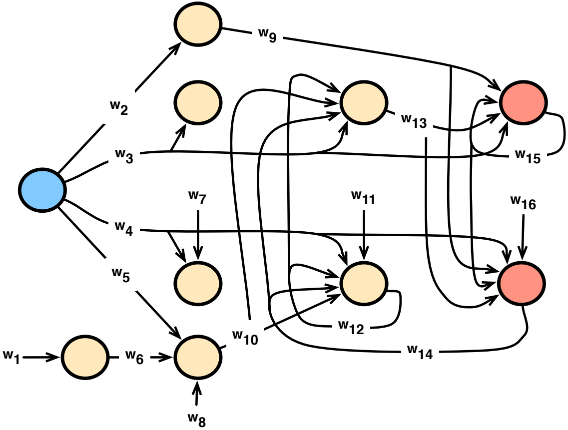
Turing Machine with Advice (TM/A)

Additional advice tape and advice function $\alpha : \mathbb{N} \longrightarrow \{0,1\}^*$

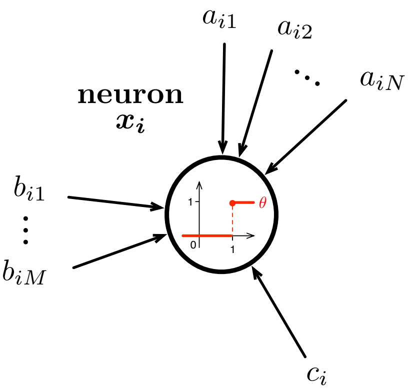


- TM/As are strictly more powerful than TMs: **P/poly** $\supsetneq$ **P**
  They are *super-Turing*...

Recurrent Neural Network

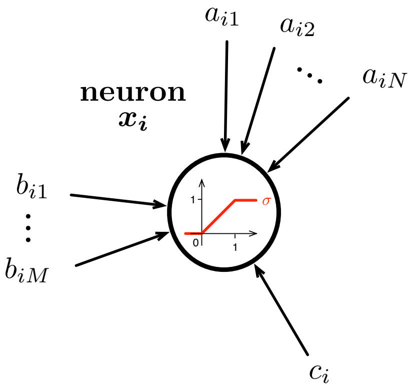


Boolean Recurrent Neural Networks



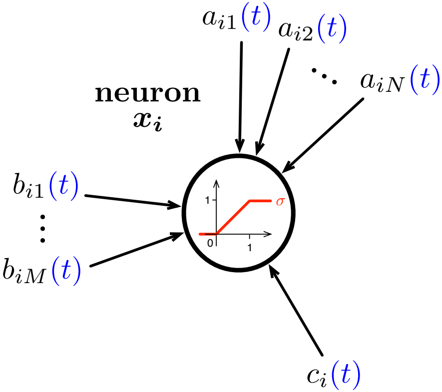
$$x_i(t+1) = \theta\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Sigmoid Recurrent Neural Networks



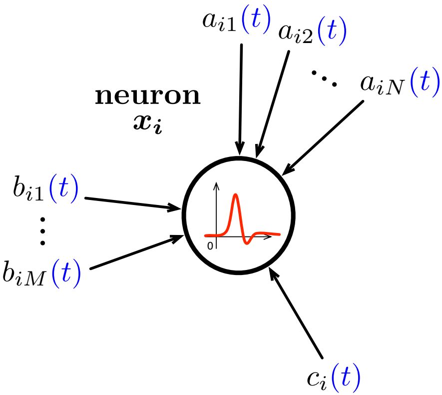
$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij} \cdot x_j(t) + \sum_{j=1}^{M} b_{ij} \cdot u_j(t) + c_i\right)$$

Evolving Recurrent Neural Networks



$$x_i(t+1) = \sigma\left(\sum_{j=1}^{N} a_{ij}(t) \cdot x_j(t) + \sum_{j=1}^{M} b_{ij}(t) \cdot u_j(t) + c_i(t)\right)$$

Bio-inspired Neural Networks



Izhikevich or Hodgkin-Huxley differential equations

Results: classical computation

	Boolean	Sigmoid		
		Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	FSA	TM	TM/poly(A)	TM/poly(A)
	REG	**P**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 95	Ca & Si 11,14	Ca & Si 11,14
$\mathbb{R}$	FSA	TM/poly(A)	TM/poly(A)	TM/poly(A)
	REG	**P/poly**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 94	Ca & Si 11,14	Ca & Si 11,14

Results: classical computation

	Boolean	Sigmoid		
		Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	FSA	TM	TM/poly(A)	TM/poly(A)
	REG	**P**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 95	Ca & Si 11,14	Ca & Si 11,14
$\mathbb{R}$	FSA	TM/poly(A)	TM/poly(A)	TM/poly(A)
	REG	**P/poly**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 94	Ca & Si 11,14	Ca & Si 11,14

Results: classical computation

	Boolean	Sigmoid		
		Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	FSA	TM	TM/poly(A)	TM/poly(A)
	REG	**P**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 95	Ca & Si 11,14	Ca & Si 11,14
$\mathbb{R}$	FSA	TM/poly(A)	TM/poly(A)	TM/poly(A)
	REG	**P/poly**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 94	Ca & Si 11,14	Ca & Si 11,14

Results: classical computation

	Boolean	Sigmoid		
		Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	FSA	TM	TM/poly(A)	TM/poly(A)
	REG	**P**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 95	Ca & Si 11,14	Ca & Si 11,14
$\mathbb{R}$	FSA	TM/poly(A)	TM/poly(A)	TM/poly(A)
	REG	**P/poly**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 94	Ca & Si 11,14	Ca & Si 11,14

Results: classical computation

	Boolean	Sigmoid		
		Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	FSA	TM	TM/poly(A)	TM/poly(A)
	REG	**P**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 95	Ca & Si 11,14	Ca & Si 11,14
$\mathbb{R}$	FSA	TM/poly(A)	TM/poly(A)	TM/poly(A)
	REG	**P/poly**	**P/poly**	**P/poly**
	Kl 56, Mi 67	Si & So 94	Ca & Si 11,14	Ca & Si 11,14

Results: infinite computation / det.

	Boolean	Sigmoid		
		Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	Muller FSA $\in BC(\Pi^0_2)$ Ca & Vi 10,14	Muller TM $= BC(\Pi^0_2)$ Ca & Vi 15,16	super-Turing $= BC(\boldsymbol{\Pi^0_2})$ Ca & Vi 15,16	super-Turing $= BC(\boldsymbol{\Pi^0_2})$ Ca & Vi 15,16
$\mathbb{R}$	Muller FSA $\in BC(\Pi^0_2)$ Ca & Vi 10,14	super-Turing $= BC(\boldsymbol{\Pi^0_2})$ Ca & Vi 15,16	super-Turing $= BC(\boldsymbol{\Pi^0_2})$ Ca & Vi 15,16	super-Turing $= BC(\boldsymbol{\Pi^0_2})$ Ca & Vi 15,16

Results: infinite computation / nondet.

	Boolean	Sigmoid		
		Static	Bi-valued Evolving	Evolving
$\mathbb{Q}$	Muller FSA $\in \Sigma^1_1$ Ca & Vi 10,14	Muller TM $= \Sigma^1_1$ Ca & Vi 15,16	super-Turing $= \boldsymbol{\Sigma}^1_1$ Ca & Vi 15,16	super-Turing $= \boldsymbol{\Sigma}^1_1$ Ca & Vi 15,16
$\mathbb{R}$	Muller FSA $\in \Sigma^1_1$ Ca & Vi 10,14	super-Turing $= \boldsymbol{\Sigma}^1_1$ Ca & Vi 15,16	super-Turing $= \boldsymbol{\Sigma}^1_1$ Ca & Vi 15,16	super-Turing $= \boldsymbol{\Sigma}^1_1$ Ca & Vi 15,16

Drawbacks of the Constructions

- Computational states of the machines are represented as Boolean states, i.e., spiking configurations of the networks.
  - ⋆ Computational states should rather be represented by *sustained activities of neural assemblies*, e.g., by *attractors*.
- Networks are not robust to cell death, synaptic plasticity, architectural plasticity in general.
  - ⋆ Networks should be robust to *architectural plasticity* and *synaptic noises*.
- We propose a novel paradigm for abstract neural computation based on *synfire rings*.

Drawbacks of the Constructions

- Computational states of the machines are represented as Boolean states, i.e., spiking configurations of the networks.

  ⋆ Computational states should rather be represented by *sustained activities of neural assemblies*, e.g., by *attractors*.

- Networks are not robust to cell death, synaptic plasticity, architectural plasticity in general.

  ⋆ Networks should be robust to *architectural plasticity* and *synaptic noises*.

- We propose a novel paradigm for abstract neural computation based on *synfire rings*.

Drawbacks of the Constructions

- Computational states of the machines are represented as Boolean states, i.e., spiking configurations of the networks.
  - ⋆ Computational states should rather be represented by *sustained activities of neural assemblies*, e.g., by *attractors*.
- Networks are not robust to cell death, synaptic plasticity, architectural plasticity in general.
  - ⋆ Networks should be robust to *architectural plasticity* and *synaptic noises*.
- We propose a novel paradigm for abstract neural computation based on *synfire rings*.

Drawbacks of the Constructions

- Computational states of the machines are represented as Boolean states, i.e., spiking configurations of the networks.

⋆ Computational states should rather be represented by *sustained activities of neural assemblies*, e.g., by *attractors*.

- Networks are not robust to cell death, synaptic plasticity, architectural plasticity in general.

⋆ Networks should be robust to *architectural plasticity* and *synaptic noises*.

- We propose a novel paradigm for abstract neural computation based on *synfire rings*.

Drawbacks of the Constructions

- Computational states of the machines are represented as Boolean states, i.e., spiking configurations of the networks.

⋆ Computational states should rather be represented by *sustained activities of neural assemblies*, e.g., by *attractors*.

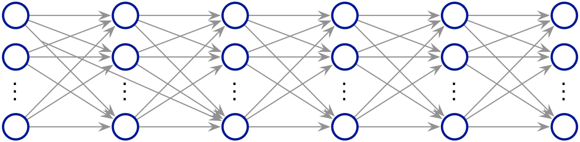
- Networks are not robust to cell death, synaptic plasticity, architectural plasticity in general.

⋆ Networks should be robust to *architectural plasticity* and *synaptic noises*.

- We propose a novel paradigm for abstract neural computation based on *synfire rings*.

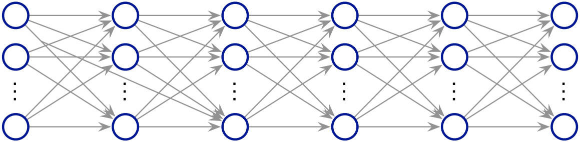
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



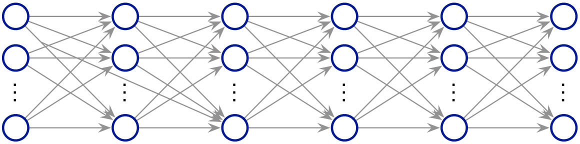
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



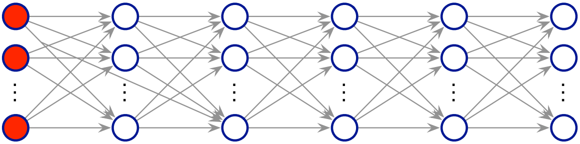
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



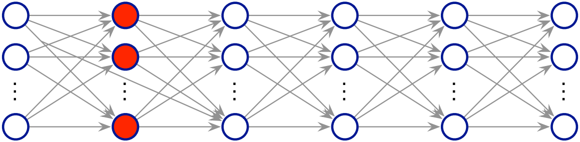
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



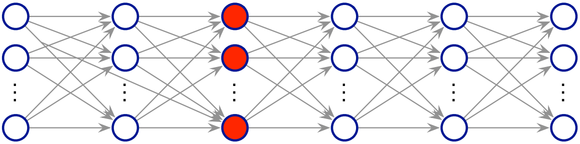
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



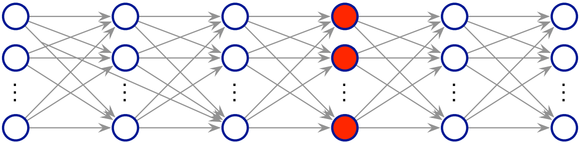
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



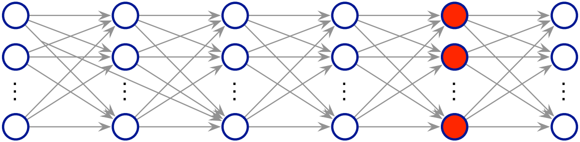
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



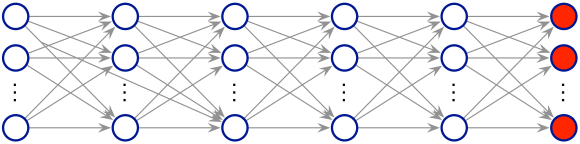
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



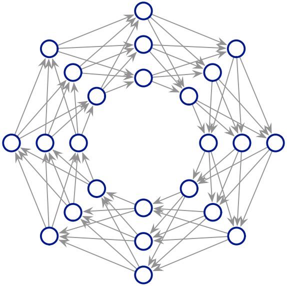
Synfire Chains

- *Synfire chains* have been theorized and experimentally validated as fundamental neuronal structures (Abeles 82).
- They allow for robust and highly precise transmission of information in neural networks.



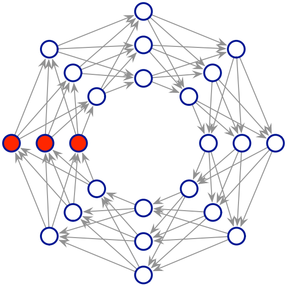
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



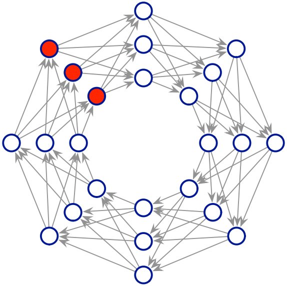
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



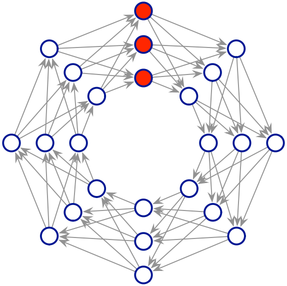
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



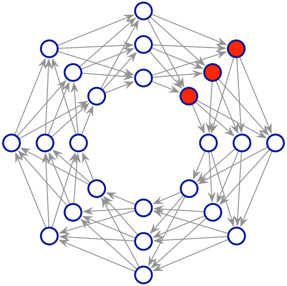
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



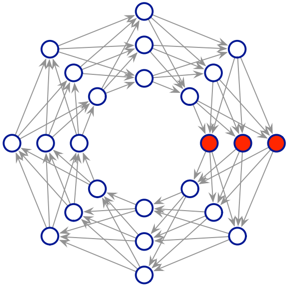
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



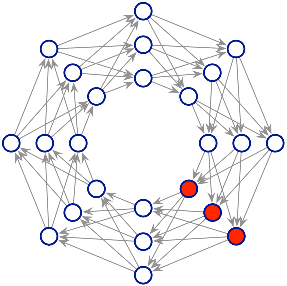
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



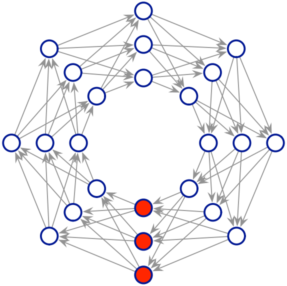
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



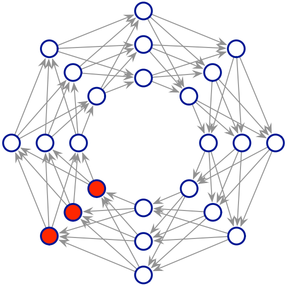
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



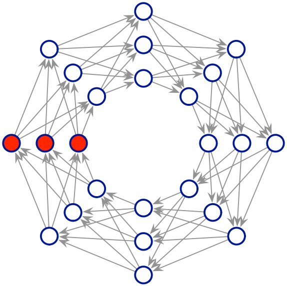
Synfire Rings

- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.

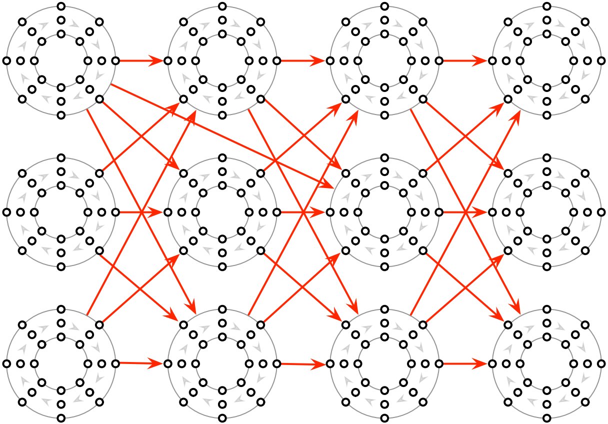


Synfire Rings

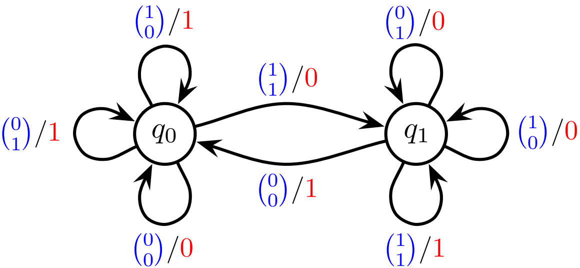
- *Synfire rings* are looping synfire chains that have been observed in self-organizing neural networks (Zheng & Triesch 14).
- *Synfire rings* allow for robust and temporally precise *self-sustained activities*.



Neural Computation with Synfire Ring Architecture

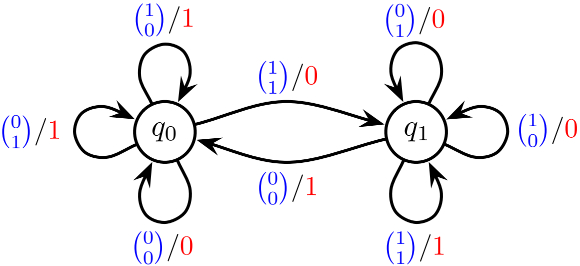


Binary Adder Automaton



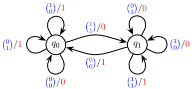
$$
\begin{array}{cccccccc}
 & {}^{1} & 1^{1} & 0^{1} & 1 & 1 & 0^{1} & 1 \\
+ & & 1 & 1 & 1 & 0 & 0 & 1 \\
\hline
 & 1 & 1 & 0 & 0 & 1 & 1 & 0
\end{array}
$$

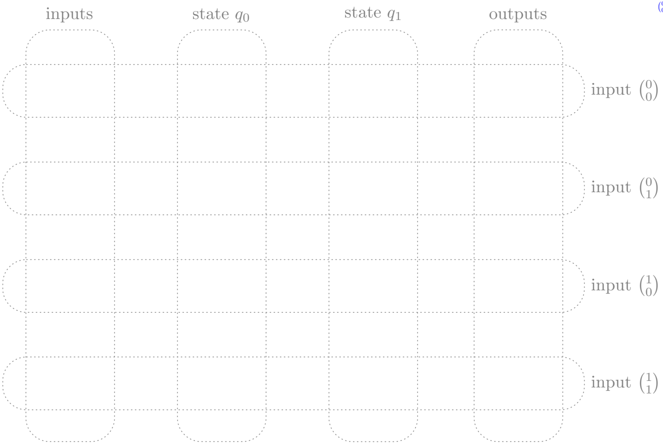
Binary Adder Automaton



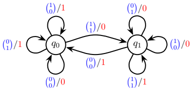
$$\begin{array}{cccccccc} & {}^{1} & 1^{1} & 0^{1} & 1 & 1 & 0^{1} & 1 \\ + & & 1 & 1 & 1 & 0 & 0 & 1 \\ \hline & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{array}$$

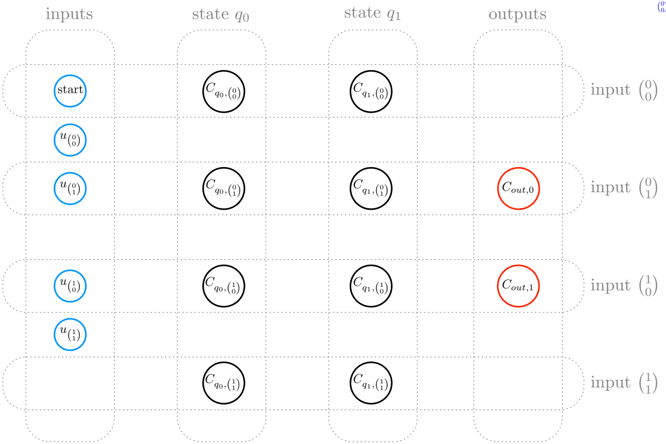
Binary Adder Boolean Neural Network



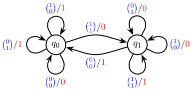


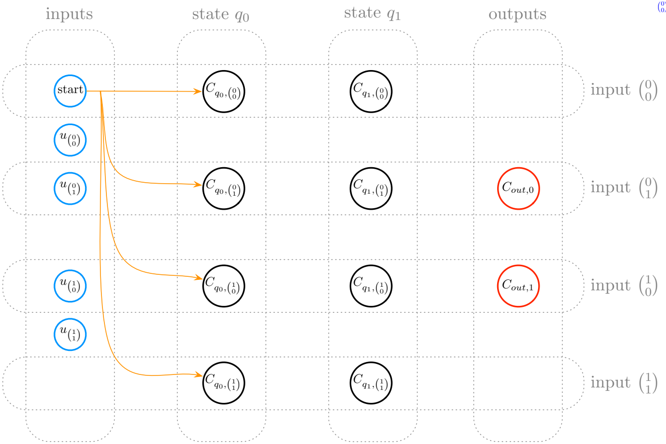
Binary Adder Boolean Neural Network



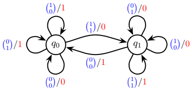


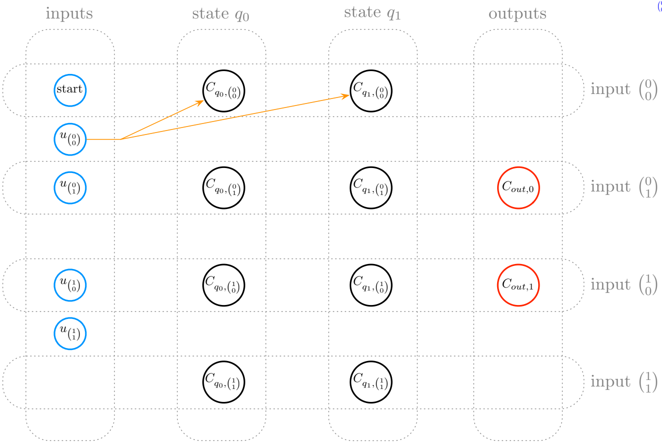
Binary Adder Boolean Neural Network



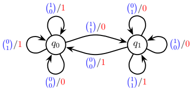


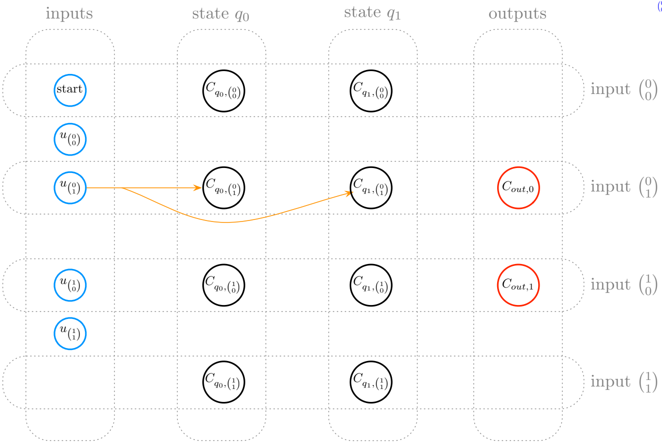
Binary Adder Boolean Neural Network



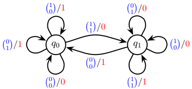


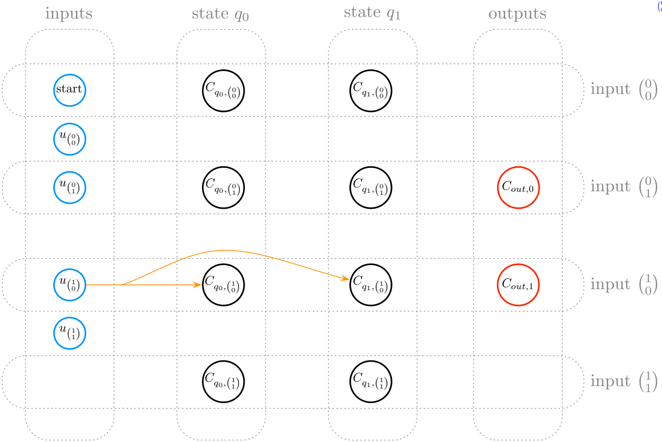
Binary Adder Boolean Neural Network



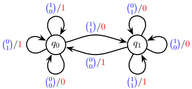


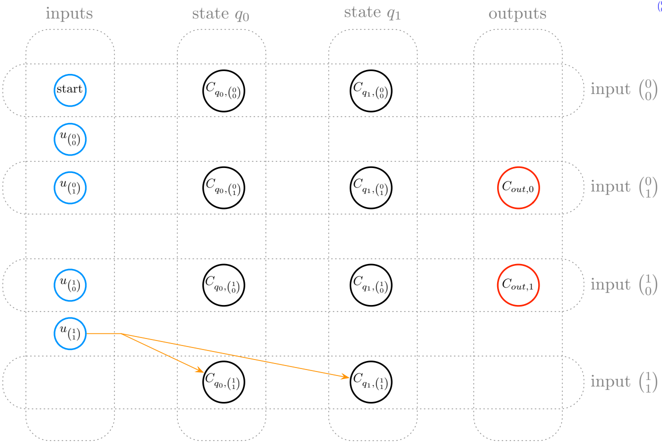
Binary Adder Boolean Neural Network



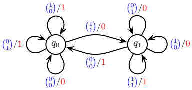


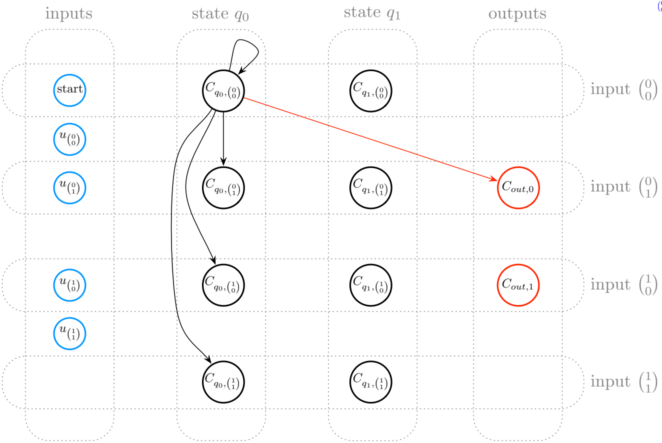
Binary Adder Boolean Neural Network



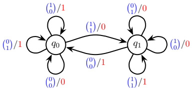


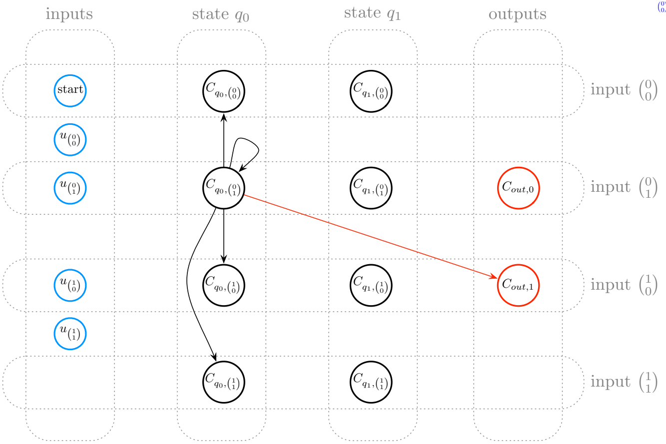
Binary Adder Boolean Neural Network



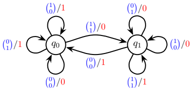


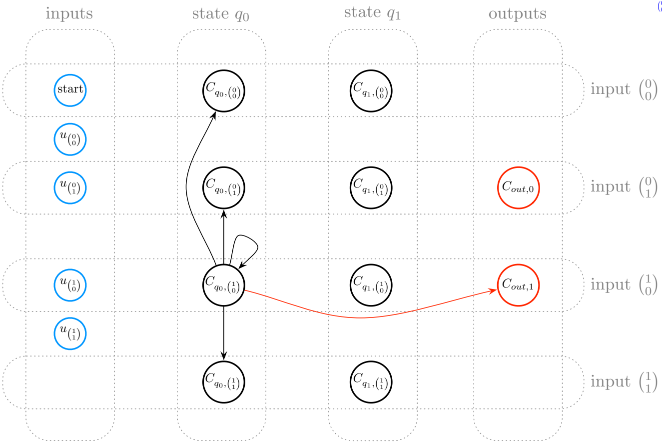
Binary Adder Boolean Neural Network



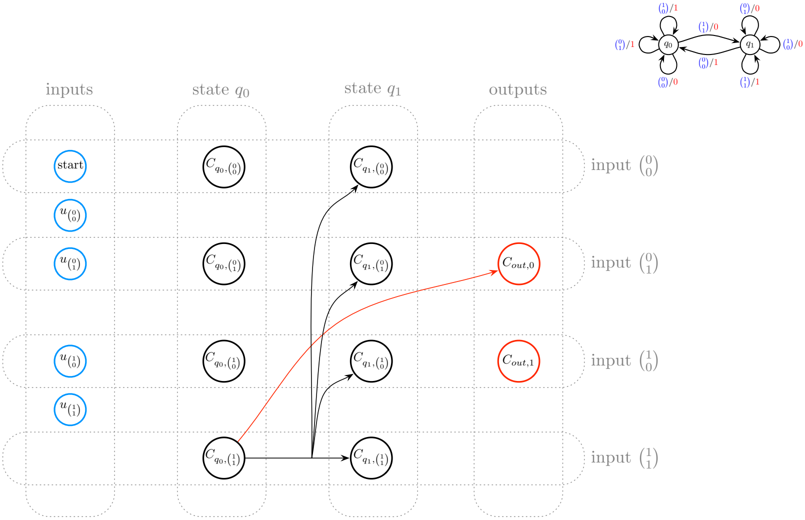


Binary Adder Boolean Neural Network

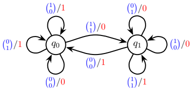


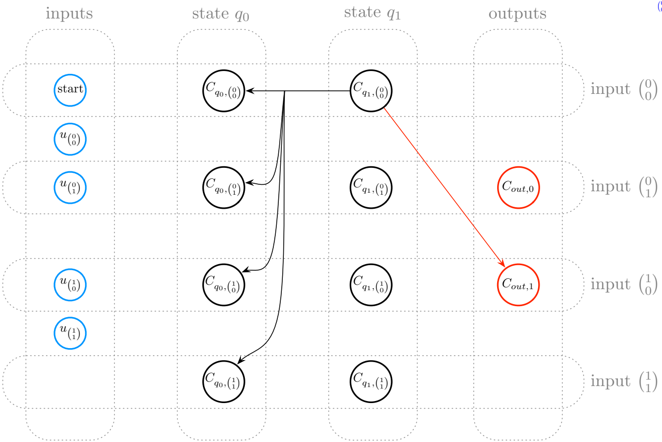


Binary Adder Boolean Neural Network

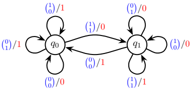


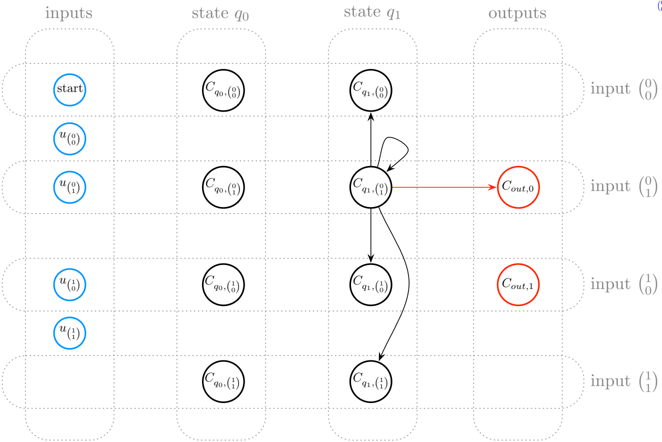
Binary Adder Boolean Neural Network



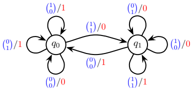


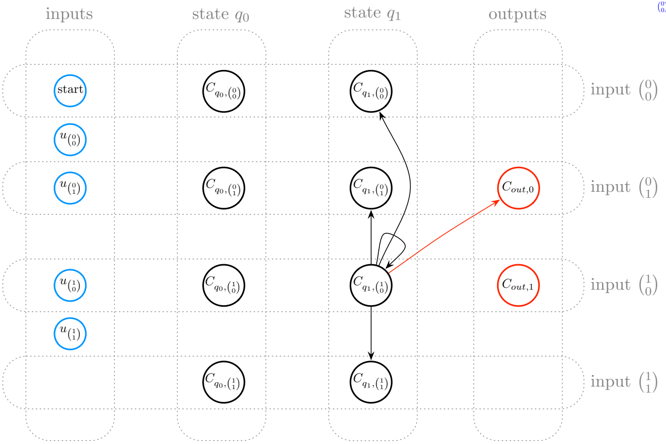
Binary Adder Boolean Neural Network



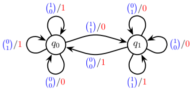


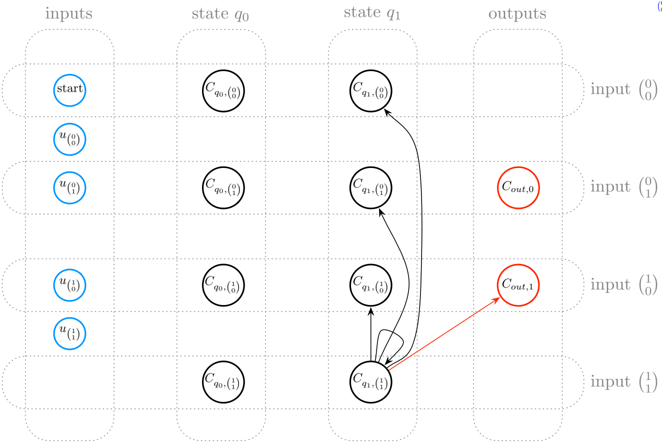
Binary Adder Boolean Neural Network



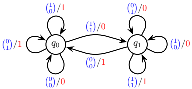


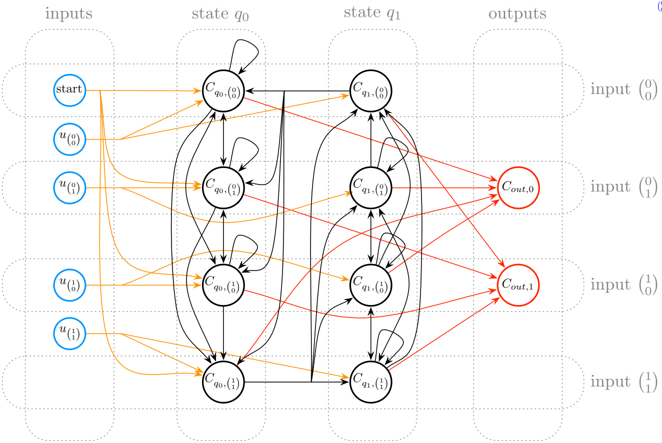
Binary Adder Boolean Neural Network



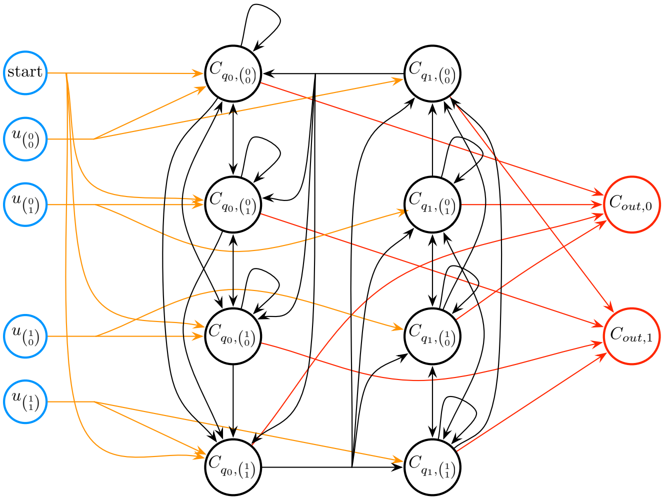


Binary Adder Boolean Neural Network

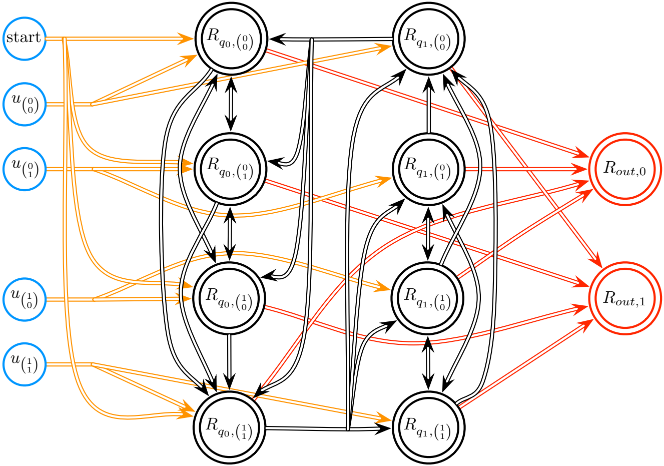




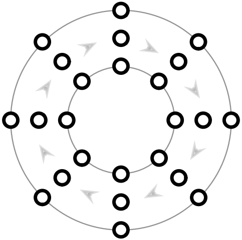
Generalization to Synfire Ring RNNs

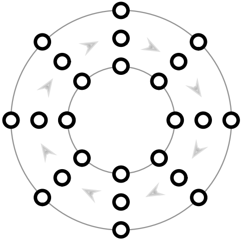


Generalization to Synfire Ring RNNs

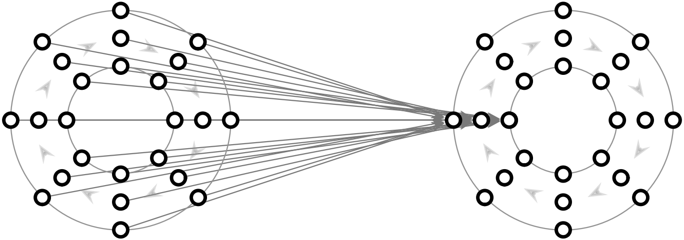


Fibres of connections & Inhibitory System

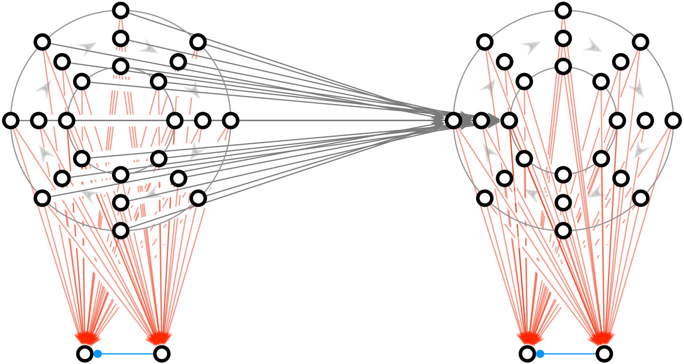




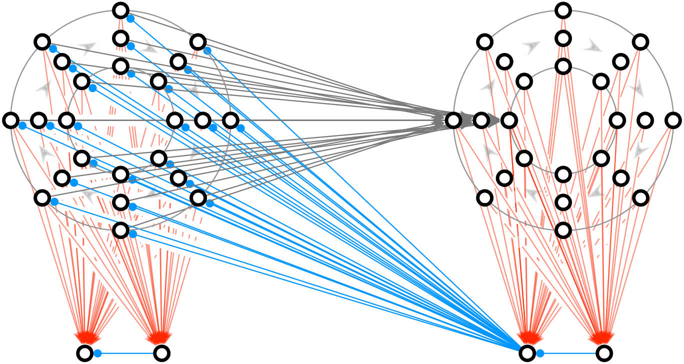
Fibres of connections & Inhibitory System



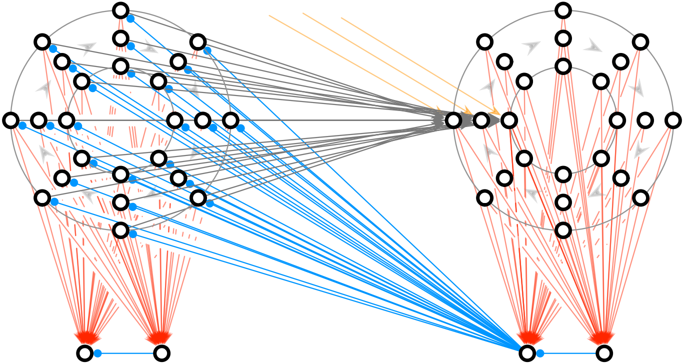
Fibres of connections & Inhibitory System



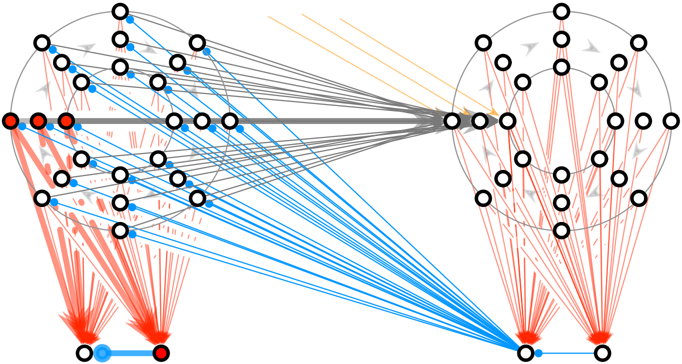
Fibres of connections & Inhibitory System



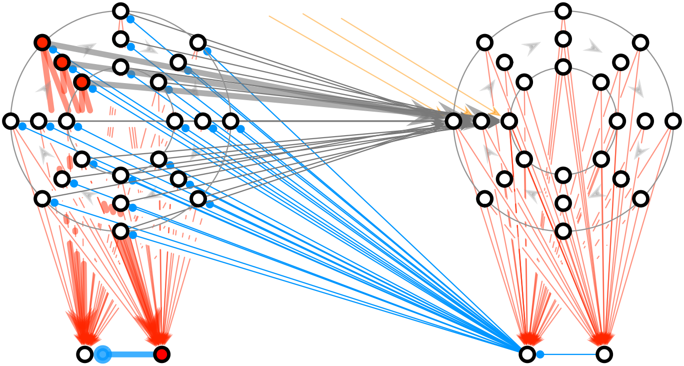
Fibres of connections & Inhibitory System



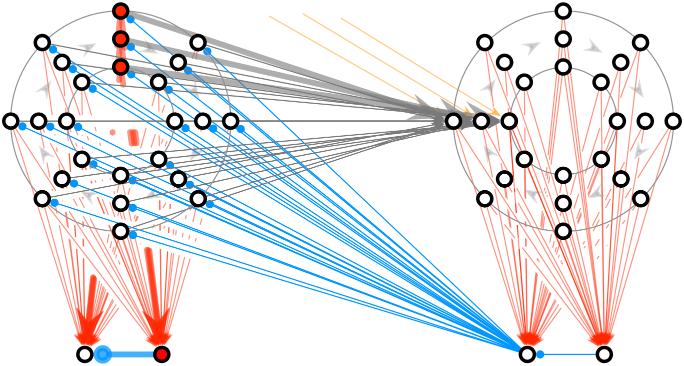
Fibres of connections & Inhibitory System



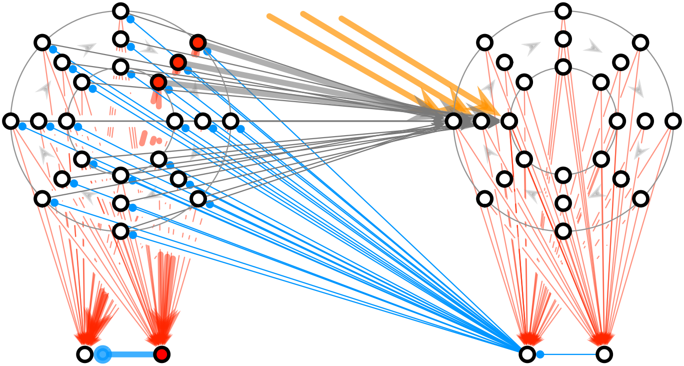
Fibres of connections & Inhibitory System



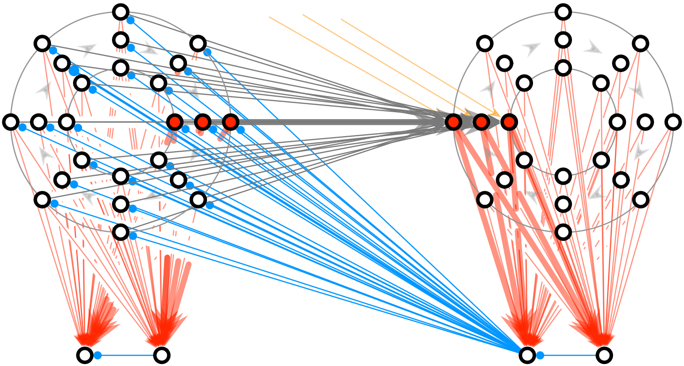
Fibres of connections & Inhibitory System



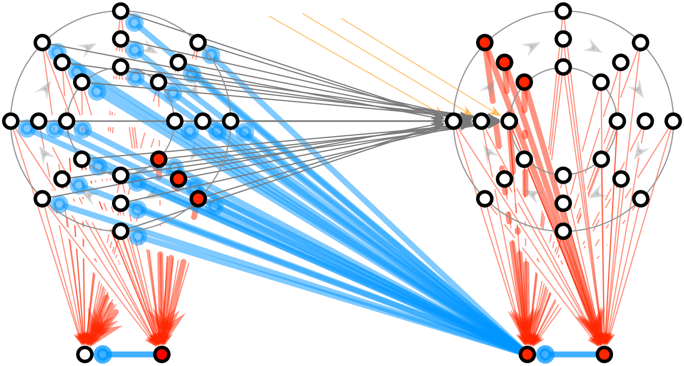
Fibres of connections & Inhibitory System



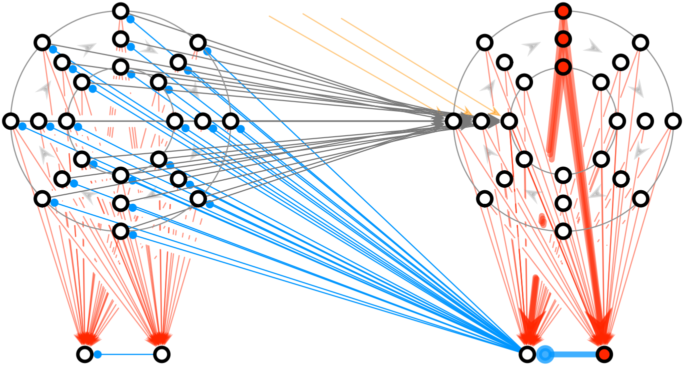
Fibres of connections & Inhibitory System



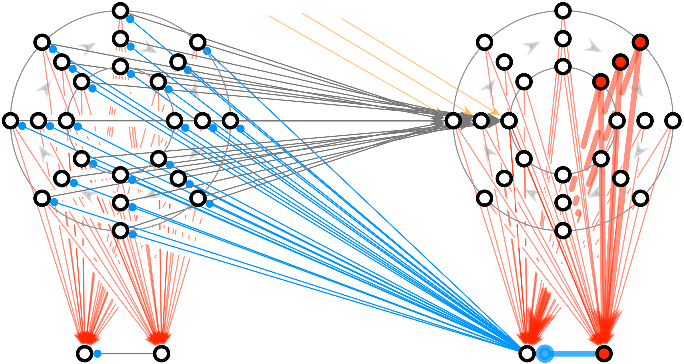
Fibres of connections & Inhibitory System



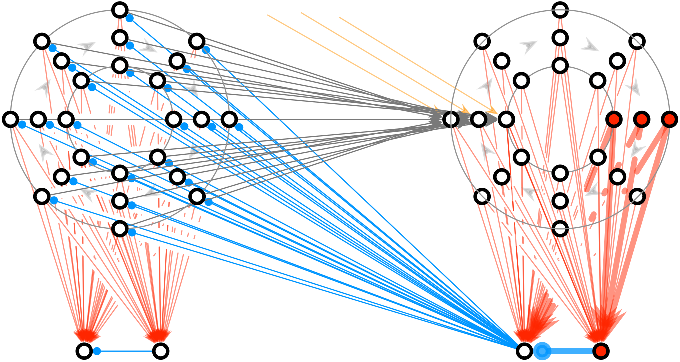
Fibres of connections & Inhibitory System



Fibres of connections & Inhibitory System



Fibres of connections & Inhibitory System



Simulation

Play movie...

Automata & Boolean RNNs with Synfire Rings

Since the construction is generic, one has the following result:

Theorem

Any finite state automaton can be simulated by some Boolean neural network composed of synfire rings.

Generalizations

- We generalize these results to the contexts of more biological neural networks:

1. Izhikevich spiking neural networks (not here)
2. Hodgkin-Huxley neural networks

Hodgkin-Huxley Neurons

$$\alpha_n(V_m) = \frac{0.01(10 - V_m)}{\exp(\frac{10-V_m}{10}) - 1} \qquad \beta_n(V_m) = 0.125\exp(\frac{-V_m}{80})$$

$$\alpha_m(V_m) = \frac{0.1(25 - V_m)}{\exp(\frac{25-V_m}{10}) - 1} \qquad \beta_m(V_m) = 4\exp(\frac{-V_m}{18})$$

$$\alpha_h(V_m) = 0.07\exp(\frac{-V_m}{20}) \qquad \beta_h(V_m) = \frac{1}{\exp(\frac{30-V_m}{10}) + 1}$$

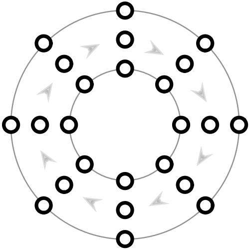
$$\frac{dn}{dt} = \alpha_n(V_m)(1-n) - \beta_n(V_m)n$$

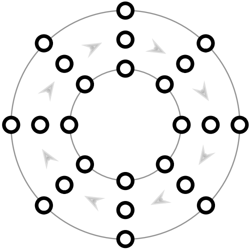
$$\frac{dm}{dt} = \alpha_m(V_m)(1-m) - \beta_m(V_m)m$$

$$\frac{dh}{dt} = \alpha_h(V_m)(1-h) - \beta_h(V_m)h$$

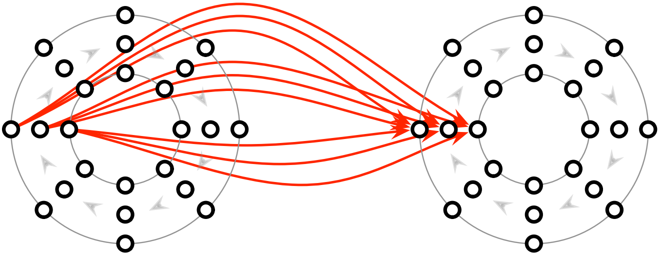
$$C_m\frac{dV_m}{dt} = I - \bar{g}_{\mathsf{K}}n^4(V_m - V_K) - \bar{g}_{\mathsf{Na}}m^3h(V_m - V_{Na}) - \bar{g}_l(V_m - V_l)$$

Fibres of connections & Inhibitory System

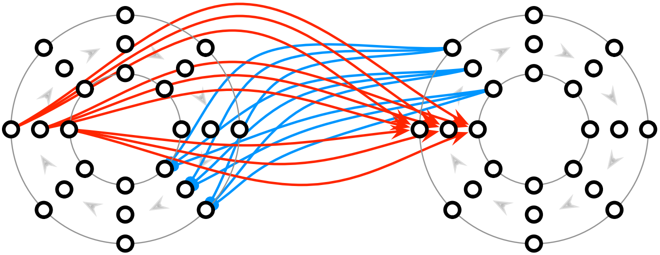




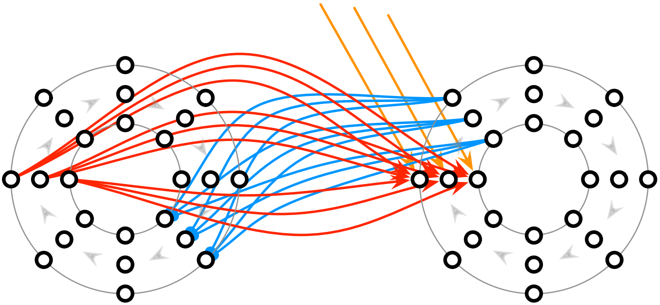
Fibres of connections & Inhibitory System



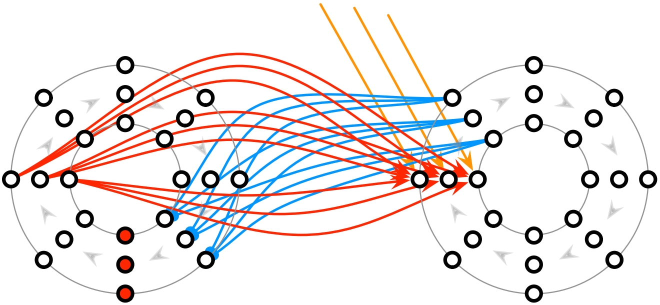
Fibres of connections & Inhibitory System



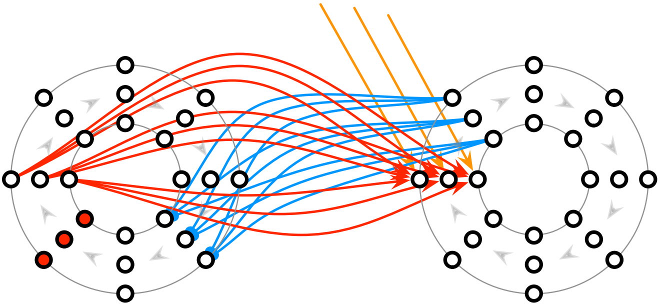
Fibres of connections & Inhibitory System



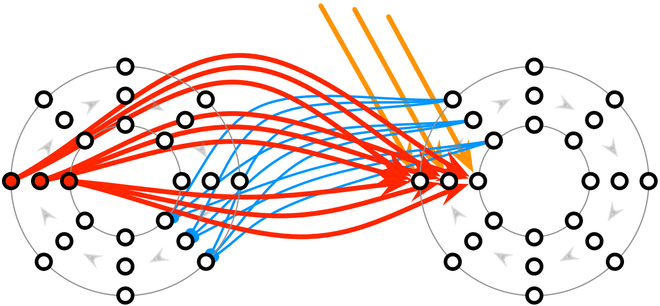
Fibres of connections & Inhibitory System



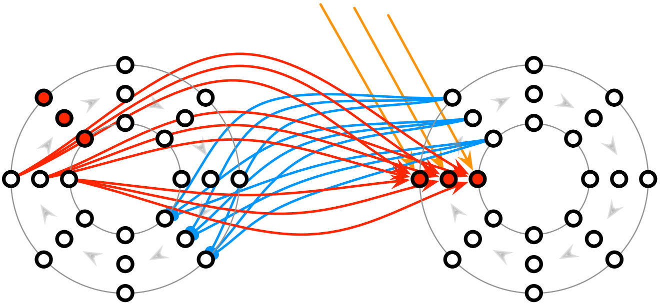
Fibres of connections & Inhibitory System



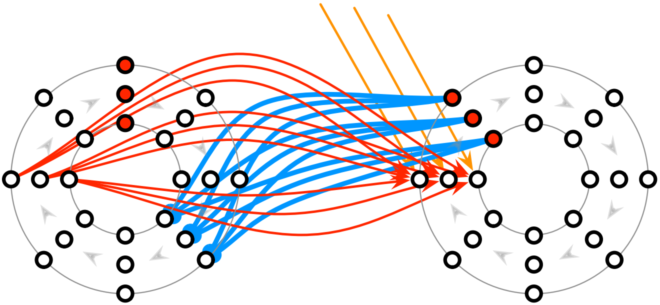
Fibres of connections & Inhibitory System



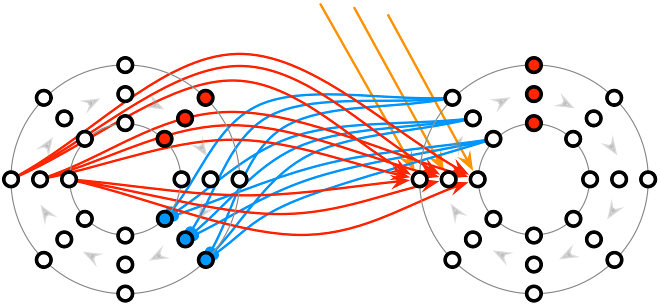
Fibres of connections & Inhibitory System



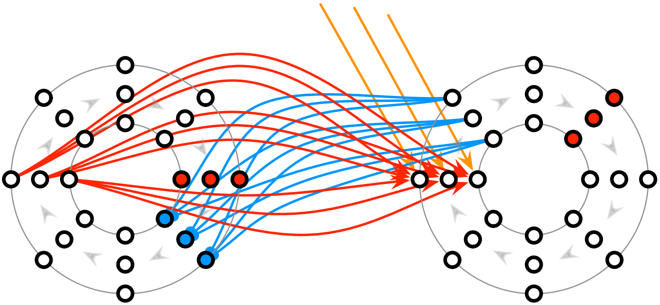
Fibres of connections & Inhibitory System



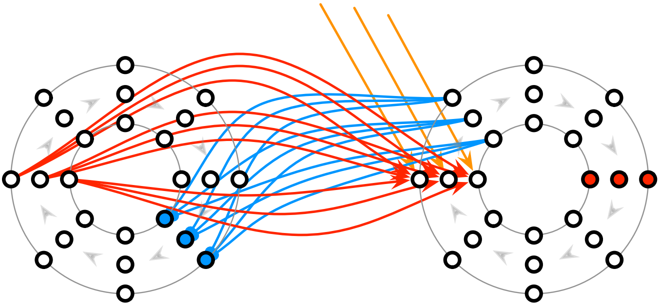
Fibres of connections & Inhibitory System



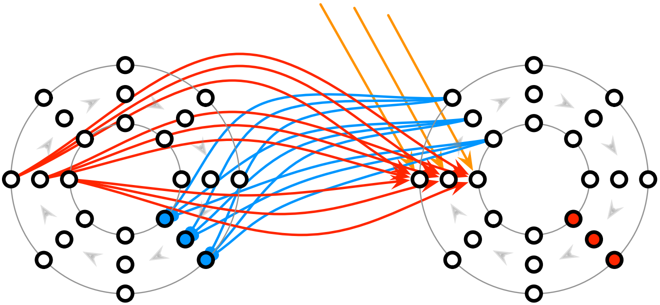
Fibres of connections & Inhibitory System



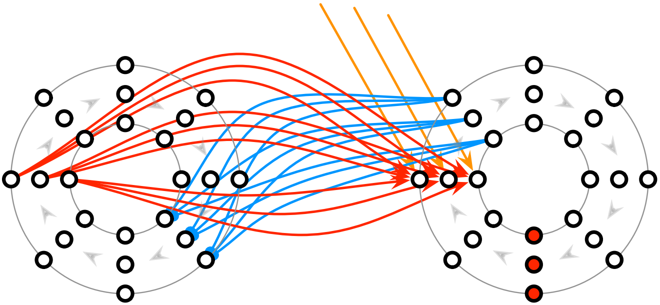
Fibres of connections & Inhibitory System



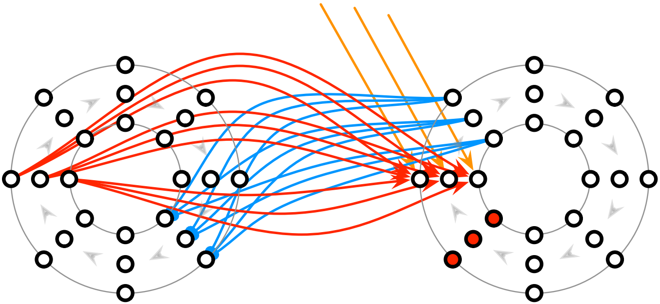
Fibres of connections & Inhibitory System



Fibres of connections & Inhibitory System



Fibres of connections & Inhibitory System



Simulation

Play movie...

Automata & Hodgkin-Huxley RNNs with Synfire Rings

Since the construction is generic, one has the following result:

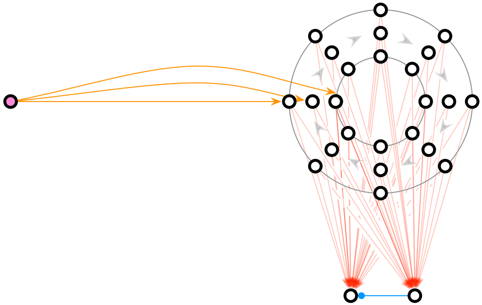
Result

Any finite state automaton can be simulated by some Hodgkin-Huxley based neural network composed of synfire rings.

Fibres of connections & Inhibitory System

We consider the following kinds of (fibres of) connections.

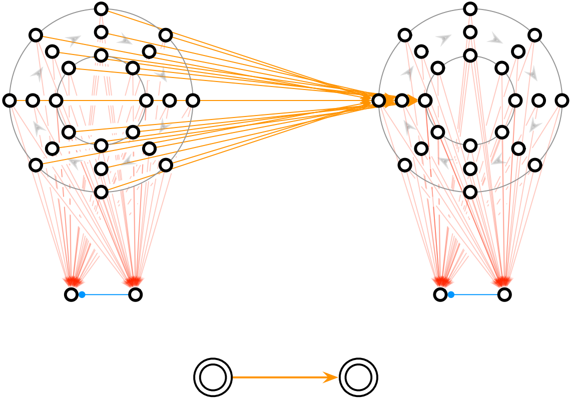
- cell to ring excitatory



Fibres of connections & Inhibitory System

We consider the following kinds of (fibres of) connections.

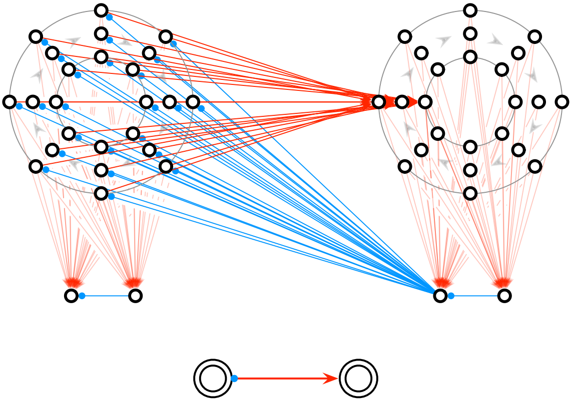
- constant excitatory



Fibres of connections & Inhibitory System

We consider the following kinds of (fibres of) connections.

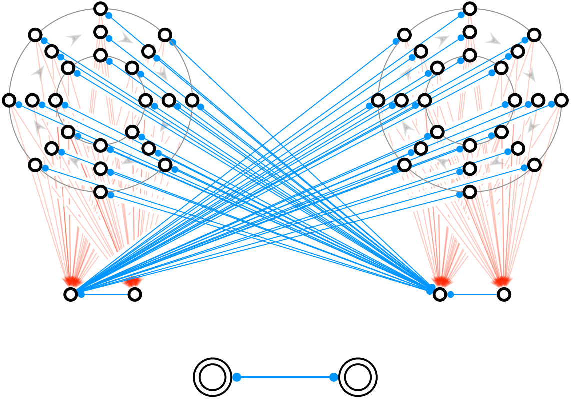
- constant excitatory / one-shot inhibitory



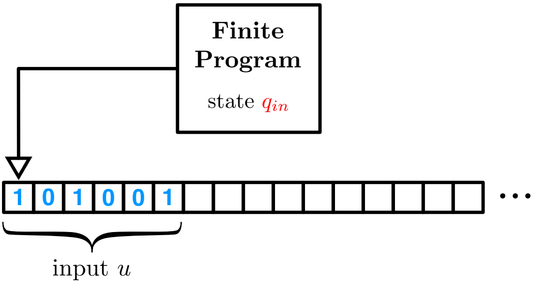
Fibres of connections & Inhibitory System

We consider the following kinds of (fibres of) connections.

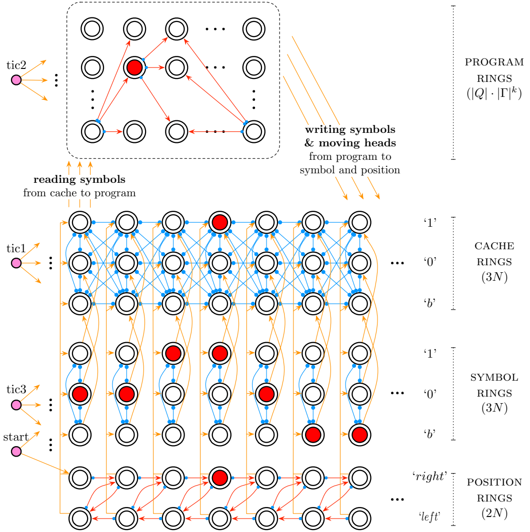
- one-shot inhibitory / one-shot inhibitory



General Architecture



General Architecture



Turing machines & Boolean RNNs with Synfire Rings

Since the construction is generic, one has the following result:

Theorem

Any bounded-space Turing machine can be simulated by some Boolean neural network composed of synfire rings.

Simulation of the bounded-space Turing machine recognizing the non-regular language $L = \{0^n 1^n 0^n : n \geq 0\}$

Play movie...

Conclusions

- We introduced a new paradigm for abstract neural computation based on the concept of synfire rings.
- Research project:

⋆ Develop bio-inspired and ML-based learning algorithms on the synfire ring architecture.

⋆ Neuromorphic implementation of these bio-inspired neural networks.

⋆ Biological implementation via cultures of neurons: TOWARDS NEURONAL COMPUTERS...

Conclusions

- We introduced a new paradigm for abstract neural computation based on the concept of synfire rings.
- Research project:
  - ⋆ Develop bio-inspired and ML-based learning algorithms on the synfire ring architecture.
  - ⋆ Neuromorphic implementation of these bio-inspired neural networks.
  - ⋆ Biological implementation via cultures of neurons: TOWARDS NEURONAL COMPUTERS...

Conclusions

- We introduced a new paradigm for abstract neural computation based on the concept of synfire rings.
- Research project:

⋆ Develop bio-inspired and ML-based learning algorithms on the synfire ring architecture.

⋆ Neuromorphic implementation of these bio-inspired neural networks.

⋆ Biological implementation via cultures of neurons: towards neuronal computers...

Conclusions

- We introduced a new paradigm for abstract neural computation based on the concept of synfire rings.
- Research project:

⋆ Develop bio-inspired and ML-based learning algorithms on the synfire ring architecture.

⋆ Neuromorphic implementation of these bio-inspired neural networks.

⋆ Biological implementation via cultures of neurons: TOWARDS NEURONAL COMPUTERS...

Conclusions

- We introduced a new paradigm for abstract neural computation based on the concept of synfire rings.
- Research project:

⋆ Develop bio-inspired and ML-based learning algorithms on the synfire ring architecture.

⋆ Neuromorphic implementation of these bio-inspired neural networks.

⋆ Biological implementation via cultures of neurons: TOWARDS NEURONAL COMPUTERS...

Conclusions

Thank you!