On the use of computational intelligence for threshold Boolean network inference with applications

Gonzalo A. Ruz[1,2,3]

[1]Complexity Research Center, UAI, Santiago, Chile

[2]Facultad de Ingeniería y Ciencias, Universidad Adolfo Ibáñez, Santiago, Chile

[3]Center of Applied Ecology and Sustainability (CAPES), Santiago, Chile

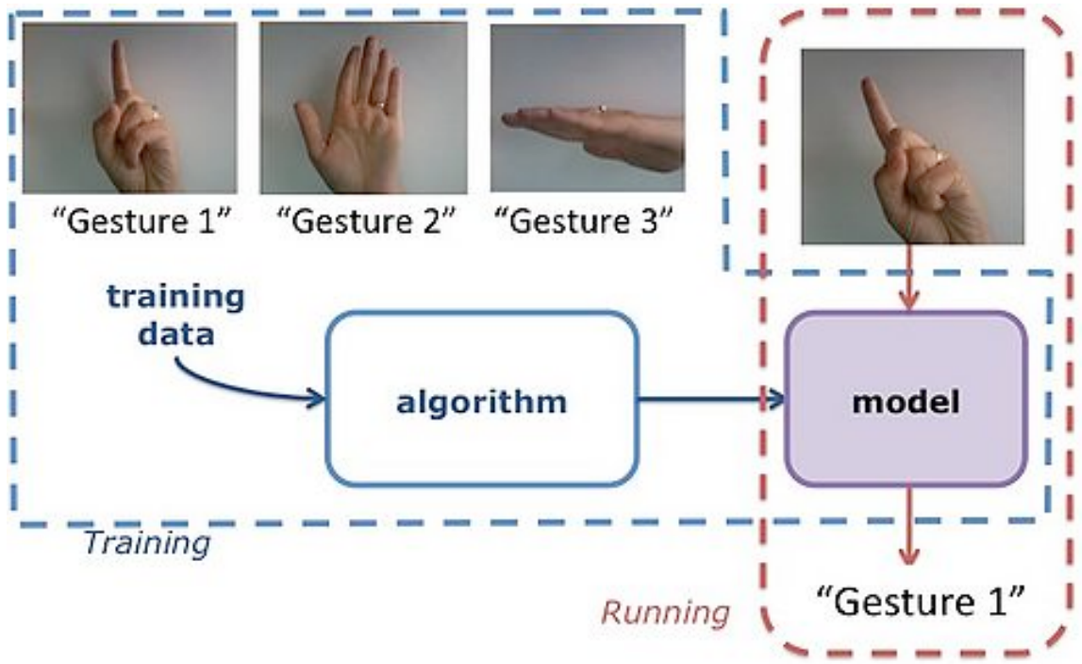
IWBN 2020, Universidad de Concepción, Chile

January 9, 2020

Outline

- Introduction
- Computational Intelligence approaches
- Application 1: Reconstruction of Boolean regulatory models of flower development
- Application 2: Inferring bistable lac operon Boolean regulatory networks
- Application 3: Boolean network model of bacterial quorum-sensing systems

Machine learning: supervised learning



Supervised learning

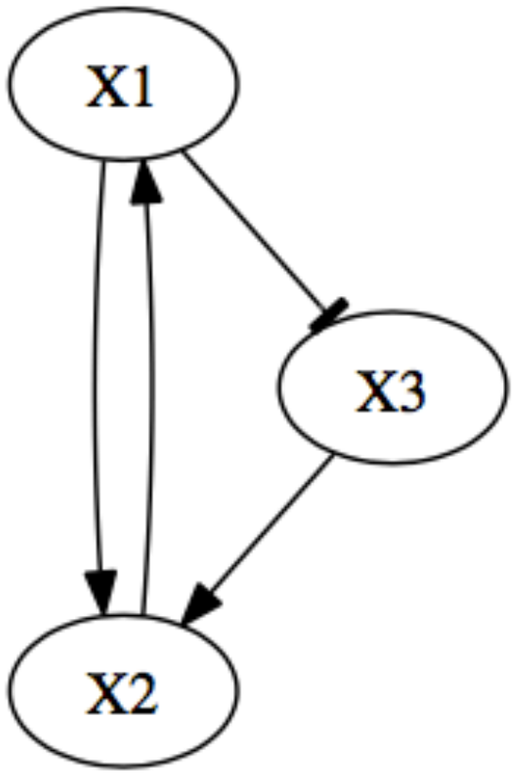
Problem Find the mapping from X to Y

Input $D = (x^1, y^1), (x^2, y^2), \ldots, (x^n, y^n)$
$f : X \times \Theta \to Y$

Output $\theta \in \Theta$, such that $f(x^i, \theta) = \hat{y}^i \approx y^i$

Boolean networks

$N = (G, F, \pi)$

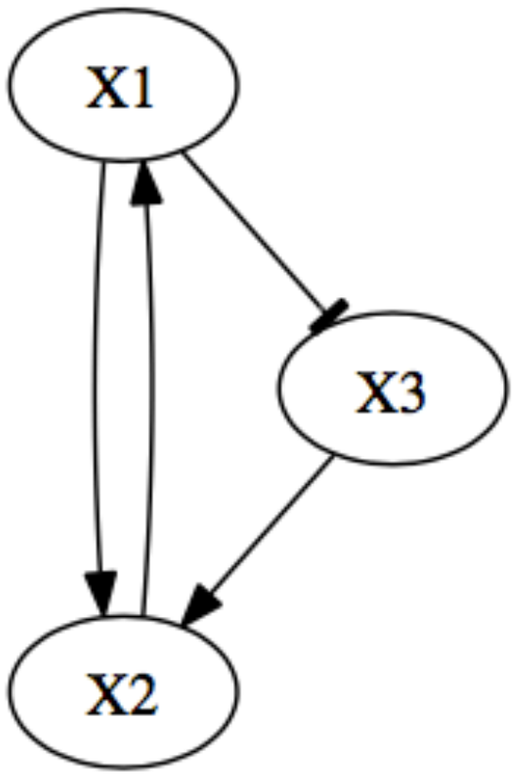


Input			Output		
$X1$	$X2$	$X3$	$X1'$	$X2'$	$X3'$
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

$X1' = X2,\ X2' = X1 \wedge X3,\ X3' = \overline{X1}$

Boolean networks

$N = (G, F, \pi)$

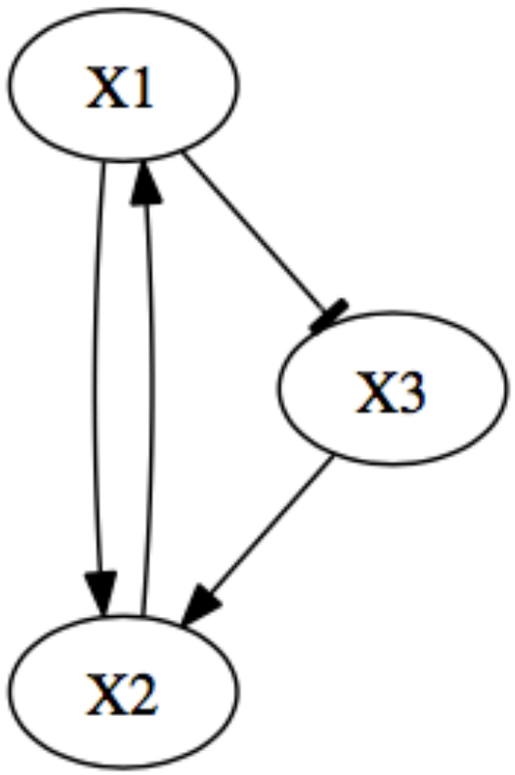


Input			Output		
$X1$	$X2$	$X3$	$X1'$	$X2'$	$X3'$
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

$X1' = X2,\ X2' = X1 \wedge X3,\ X3' = \overline{X1}$

Boolean networks

$N = (G, F, \pi)$

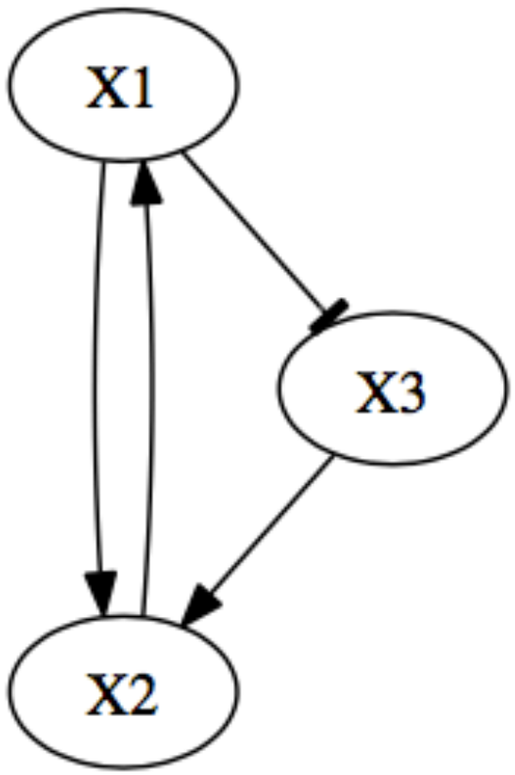


Input			Output		
$X1$	$X2$	$X3$	$X1'$	$X2'$	$X3'$
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

$X1' = X2,\ X2' = X1 \wedge X3,\ X3' = \overline{X1}$

Boolean networks

$N = (G, F, \pi)$

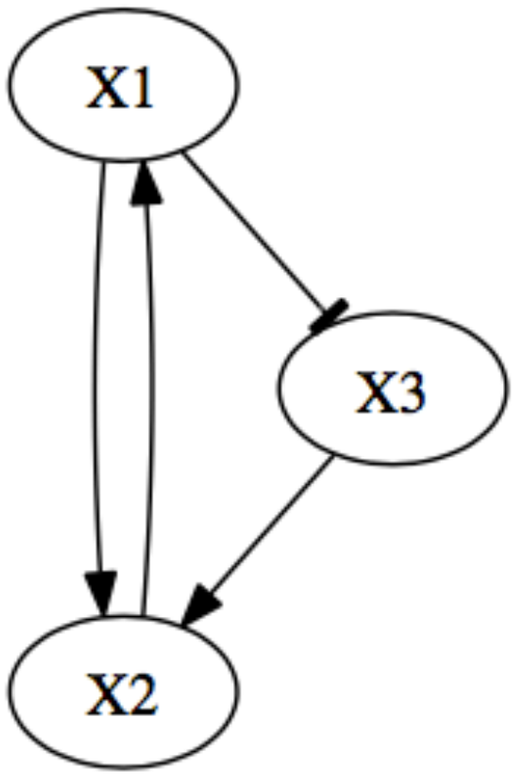


Input			Output		
$X1$	$X2$	$X3$	$X1'$	$X2'$	$X3'$
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

$X1' = X2,\ X2' = X1 \wedge X3,\ X3' = \overline{X1}$

Boolean networks

$N = (G, F, \pi)$

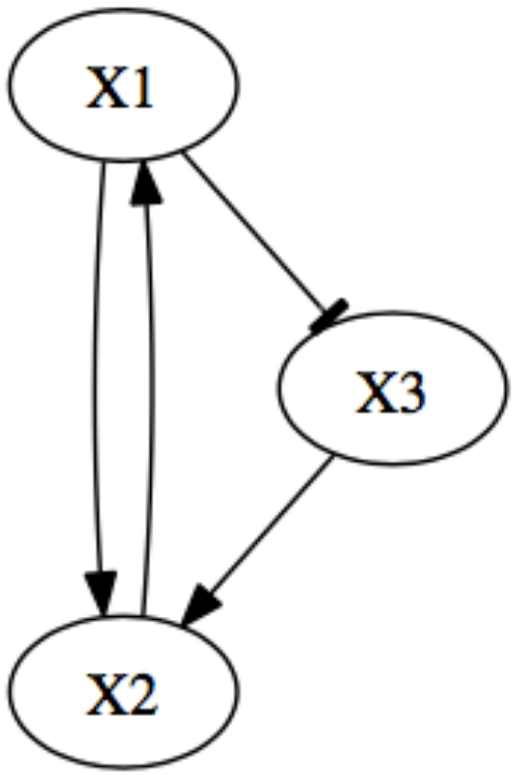


Input			Output		
$X1$	$X2$	$X3$	$X1'$	$X2'$	$X3'$
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

$X1' = X2,\ X2' = X1 \wedge X3,\ X3' = \overline{X1}$

Boolean networks

$N = (G, F, \pi)$

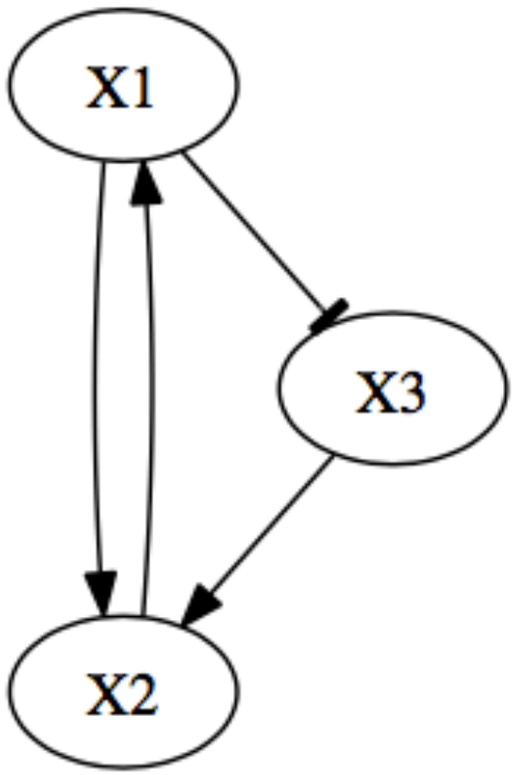


Input			Output		
$X1$	$X2$	$X3$	$X1'$	$X2'$	$X3'$
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

$X1' = X2,\ X2' = X1 \wedge X3,\ X3' = \overline{X1}$

Boolean networks

$N = (G, F, \pi)$



Input			Output		
$X1$	$X2$	$X3$	$X1'$	$X2'$	$X3'$
0	0	0	0	0	1
0	0	1	0	0	1
0	1	0	1	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	0
1	1	0	1	0	0
1	1	1	1	1	0

$X1' = X2,\ X2' = X1 \wedge X3,\ X3' = \overline{X1}$

Threshold Boolean networks

- Updates of each node in the network are computed by

$$x_i(t+1) = H\left(\sum_{j=1}^{n} w_{ij} x_j(t) - \theta_i\right)$$

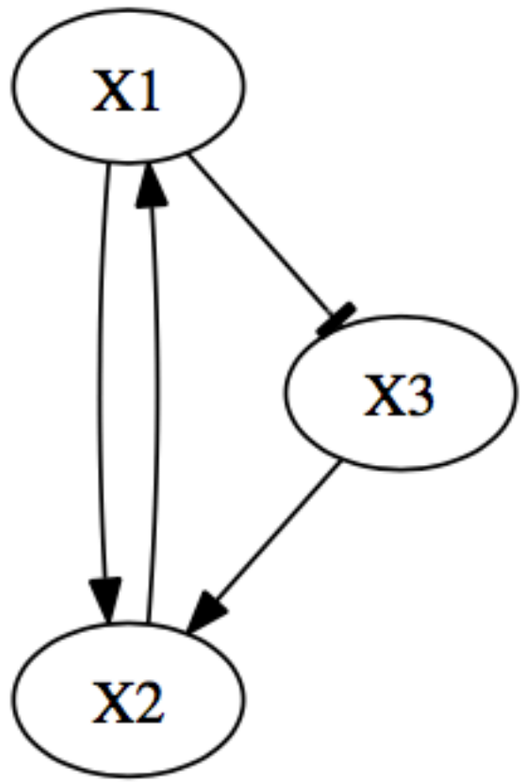
$$H(z) = \begin{cases} 1, & \text{if } z \geq 0 \\ 0, & \text{if } z < 0 \end{cases}$$

- With ω_{ij} the weight of the edge coming from node j into node i, and θ_i the activation threshold of node i.

Threshold Boolean network version of the previous example

$$W = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ -1 & 0 & 0 \end{pmatrix}$$

$$\Theta = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$$



Computational intelligence approaches

Computational intelligence approaches

- **Evolutionary computation**

Computational intelligence approaches

- **Evolutionary computation**
- **Neural networks**

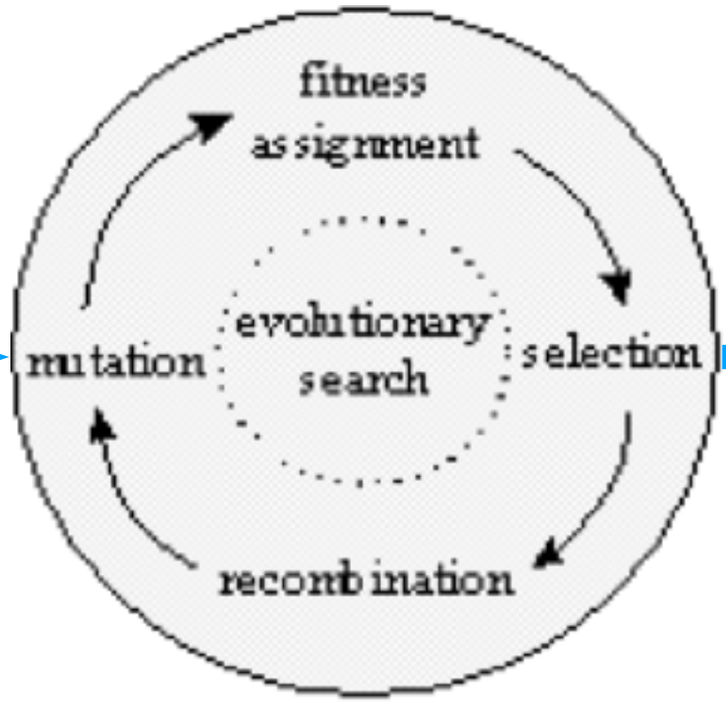
Computational intelligence approaches

- **Evolutionary computation**
- **Neural networks**
- Fuzzy logic

Evolutionary computation

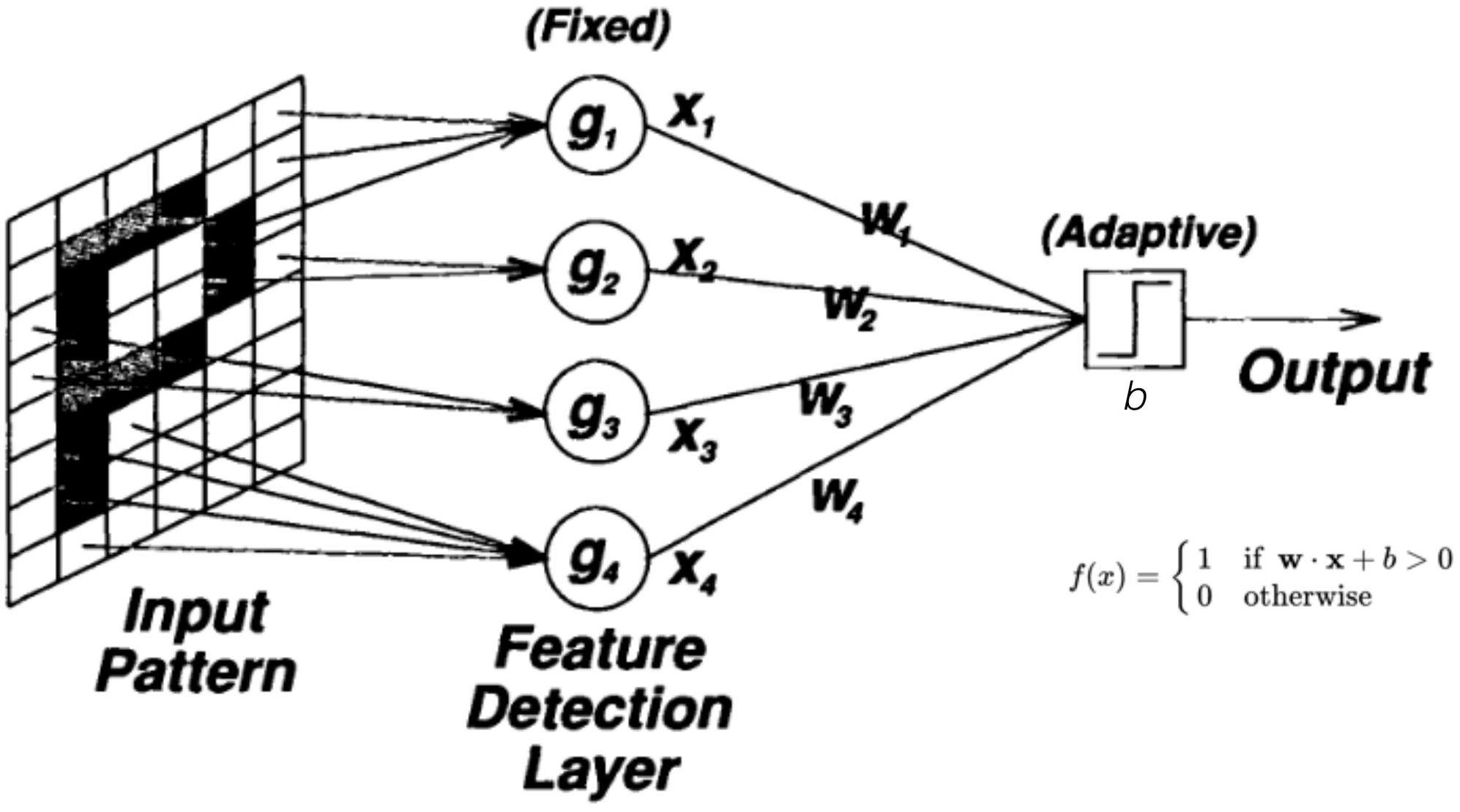
Problem

Coding of solutions
Fitness function
Evolutionary operators
Domain knowledge



Solution

Neural networks

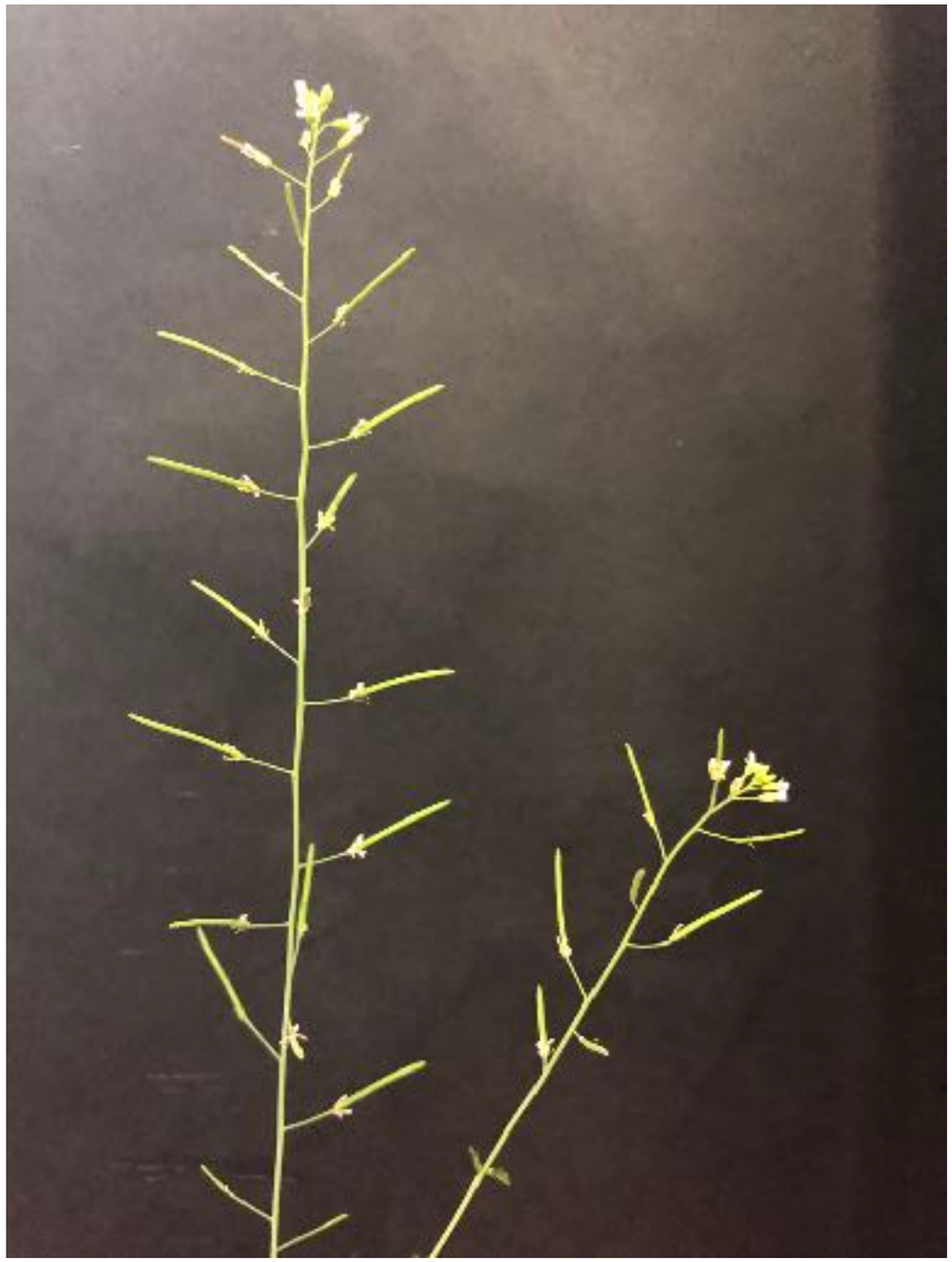


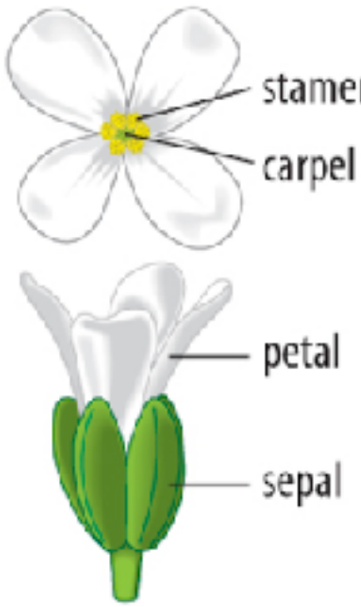
The Perceptron

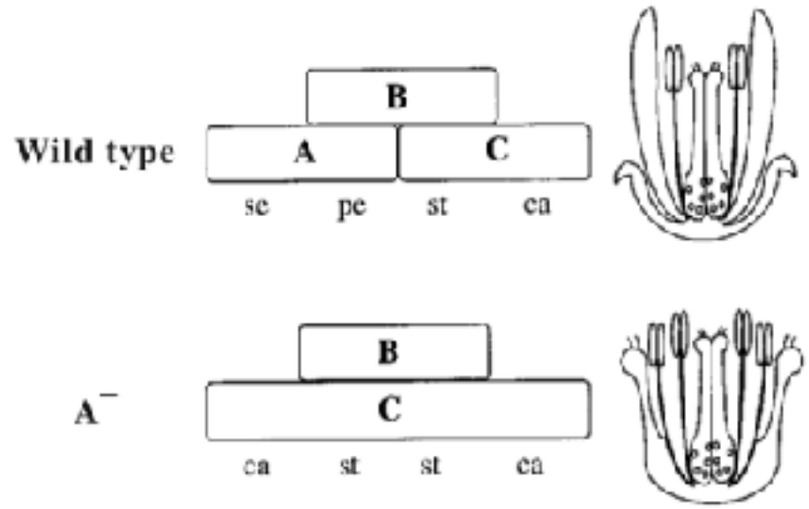
Application 1

Ruz, G.A., Goles, E., Sené, S. Reconstruction of Boolean regulatory models of flower development exploiting an evolution strategy, The 2018 IEEE Congress on Evolutionary Computation (IEEE CEC 2018), Rio de Janeiro, Brazil, July 8-13, 2018, pp. 1-8.

Flower development

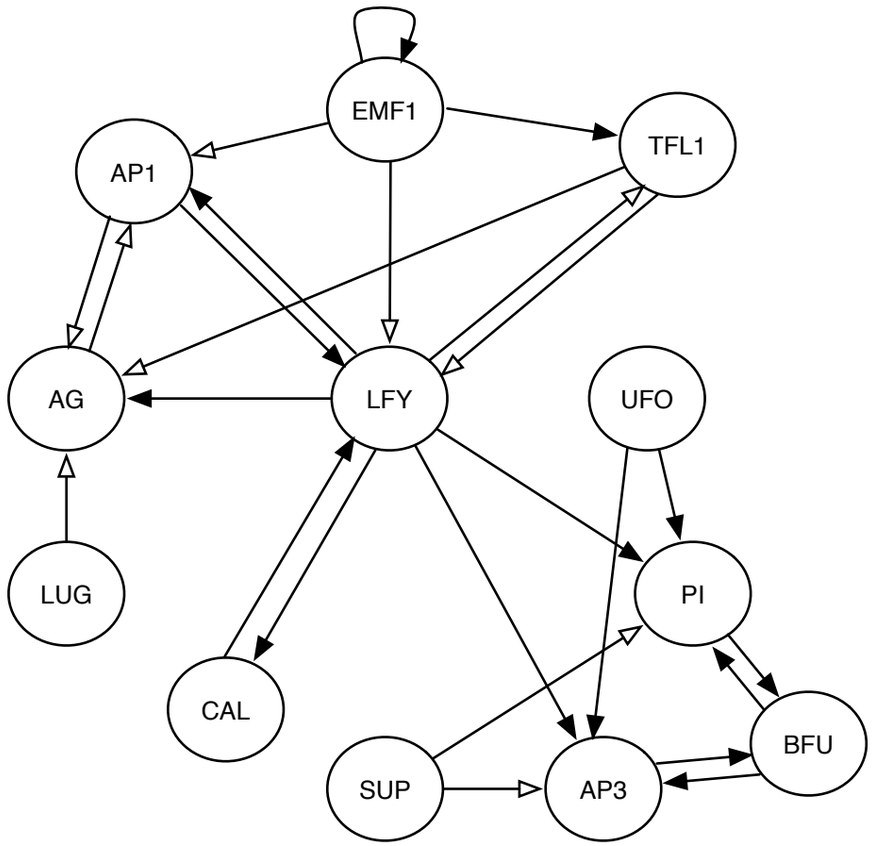






L. Mendoza and E. R. Alvarez-Buylla, “Dynamics of the genetic regulatory network for arabidopsis thaliana flower morphogenesis,” Journal of Theoretical Biology, vol. 193, pp. 307–319, 1998.

Original Mendoza & Alvarez-Buylla network



$$x_i(t+1) = H\left(\sum_{j=1}^{n} w_{ij}x_j(t) - \theta_i\right)$$

$$H(z) = \begin{cases} 1, & \text{if } z > 0 \\ 0, & \text{if } z \leq 0 \end{cases}$$

$$W = \begin{array}{c} \\ EMF1 \\ TFL1 \\ LFY \\ AP1 \\ CAL \\ LUG \\ UFO \\ BFU \\ AG \\ AP3 \\ PI \\ SUP \end{array} \begin{array}{c} \begin{array}{cccccccccccc} EMF1 & TFL1 & LFY & AP1 & CAL & LUG & UFO & BFU & AG & AP3 & PI & SUP \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & -2 & 1 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & -2 \\ 0 & 0 & 4 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \quad \Theta = \begin{pmatrix} 0 \\ 0 \\ 3 \\ -1 \\ 1 \\ 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

25 edges =
15 positive edges +
10 negative edges

Attractors

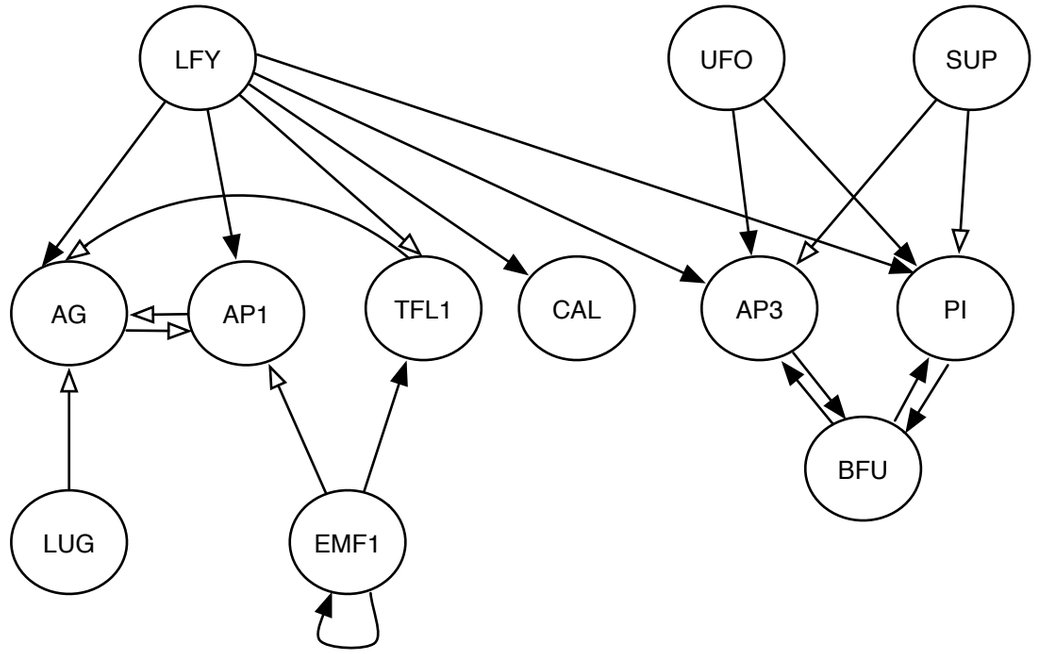
Attractors	Sequential	Parallel	Cell types
Fixed point 1	000100000000	000100000000	*Sep*
Fixed point 2	000100010110	000100010110	*Pet*
Fixed point 3	000000001000	000000001000	*Car*
Fixed point 4	000000011110	000000011110	*Sta*
Fixed point 5	110000000000	110000000000	*Inf*
Fixed point 6	110000010110	110000010110	*Mut*
Limit cycle 1	–	000100010000 000100000110	None
Limit cycle 2	–	000000000000 000100001000	None
Limit cycle 3	–	000000010000 000100001110	None
Limit cycle 4	–	000000000110 000100011000	None
Limit cycle 5	–	000000010110 000100011110	None
Limit cycle 6	–	000000001110 000000011000	None
Limit cycle 7	–	110000000110 110000010000	None

Attractors of the original Mendoza & Alvarez-Buylla network dynamics for the sequential and parallel iteration modes and the corresponding cell types. In the descriptions of each configuration, genes are ordered as follows: EMF1, TFL1, LFY, AP1, CAL, LUG, UFO, BFU, AG, AP3, PI, SUP.

doi:10.1371/journal.pone.0011793.t001

Demongeot J, Goles E, Morvan M, Noual M, Sene´ S (2010) Attraction Basins as Gauges of Robustness against Boundary Conditions in Biological Complex Systems. PLoS ONE 5(8): e11793.

Reduced Mendoza & Alvarez-Buylla network



21 edges =
13 positive edges +
8 negative edges

$$W = \begin{pmatrix} & EMF1 & TFL1 & LFY & AP1 & CAL & LUG & UFO & BFU & AG & AP3 & PI & SUP \\ EMF1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ TFL1 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ LFY & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ AP1 & -2 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 \\ CAL & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ LUG & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ UFO & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ BFU & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ AG & 0 & -2 & 1 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ AP3 & 0 & 0 & 3 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & -2 \\ PI & 0 & 0 & 4 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & -1 \\ SUP & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \Theta = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -2 \\ 1 \\ 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Demongeot J, Goles E, Morvan M, Noual M, Sene´ S (2010) Attraction Basins as Gauges of Robustness against Boundary Conditions in Biological Complex Systems. PLoS ONE 5(8): e11793.

Problem description

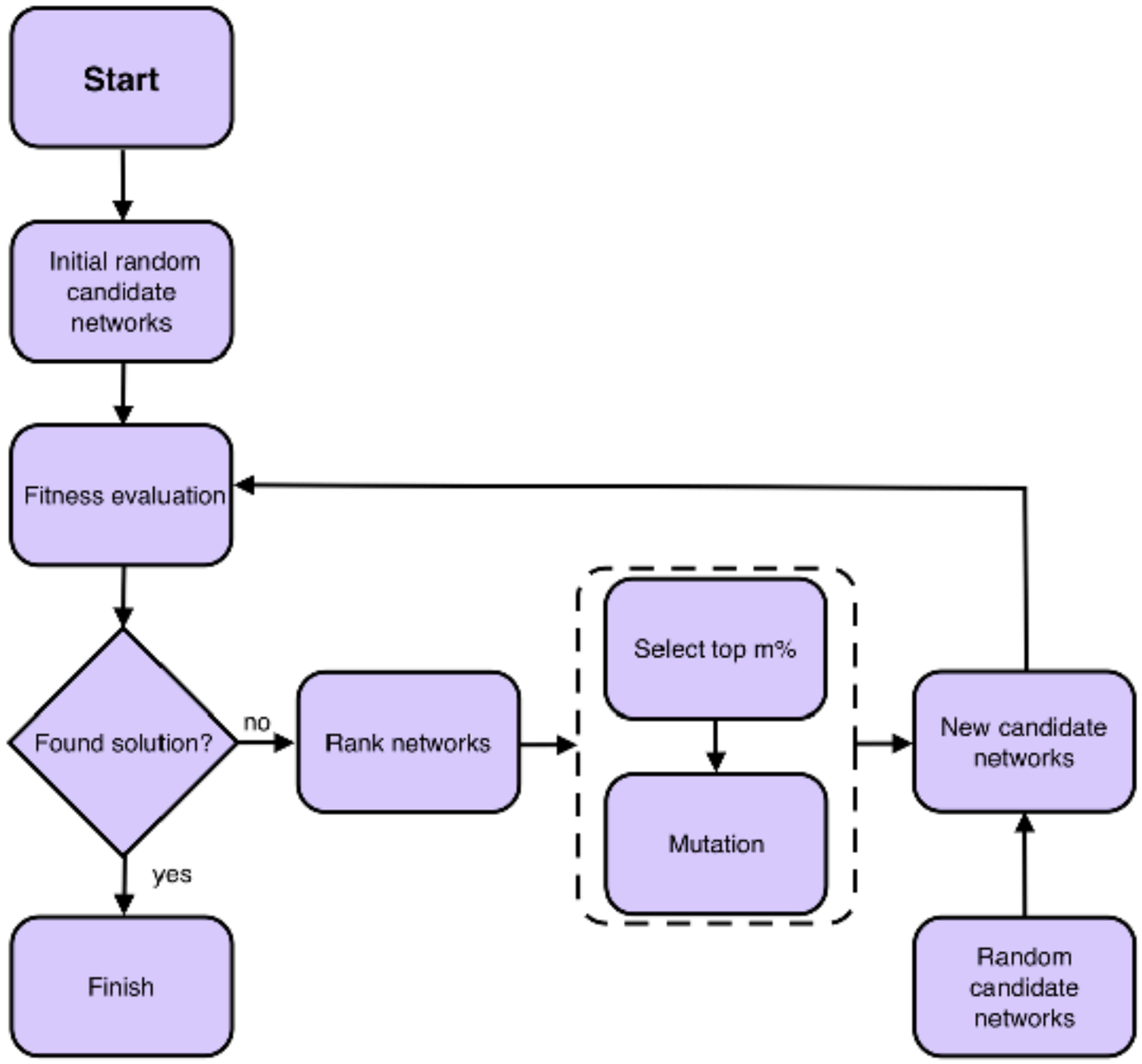
- Reconstruction of synthetic gene regulatory networks of flower development in *Arabidopsis thaliana* under the threshold Boolean network formalism.

- We use the Mendoza & Alvarez-Buylla network and the reduced model as the starting point, employing an evolution strategy to find the different network parameters (weight matrix and threshold vector) that yield synthetic networks with the same attractors as the original model.

- We analyze topological (wiring) and dynamical characteristics of the resulting networks. Also, we are interested to see if networks with fewer edges than the reduced model can be found.

- Overall, the possibility to explore neighboring solutions around the original and reduced model will allow us to shed light on how robust, in the sense of the network structure, are these models.

Evolution strategy flow chart to search for synthetic networks



Simulations

- **Simulation 1**: Infer 1000 synthetic networks that contain only the six fixed points of the original network. We use the original network as the seed to generate candidate solutions. For the resulting networks, we will compute the distribution of the total, positive, and negative number of edges.

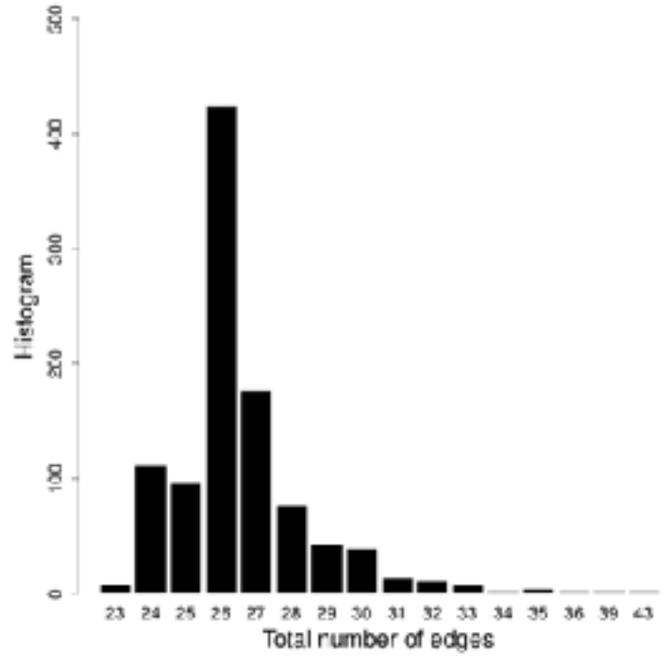
- **Simulation 2**: The same as Simulation 1, but now using the reduced network as the seed.

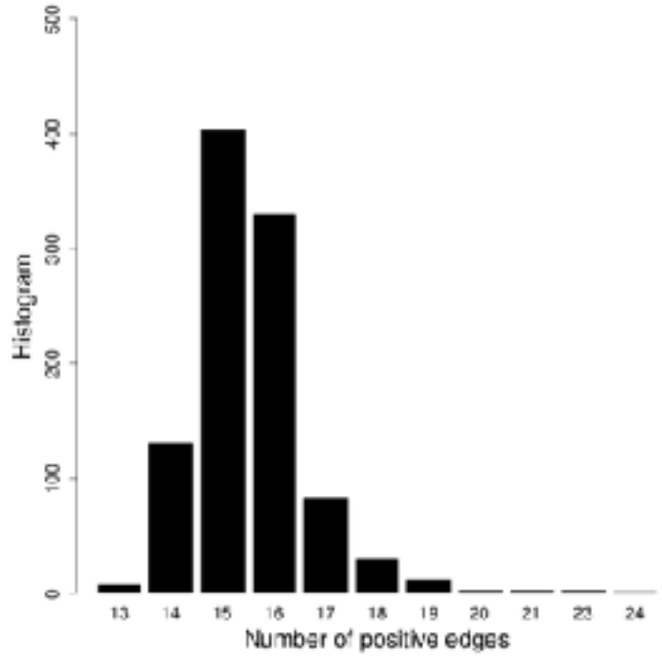
- **Simulation 3**: The same as Simulation 1, but now we will reconstruct synthetic networks that have the first four fixed points of the original network, that are associated to specific cell types of the flower: sepal, petal, carpel, and stamen.

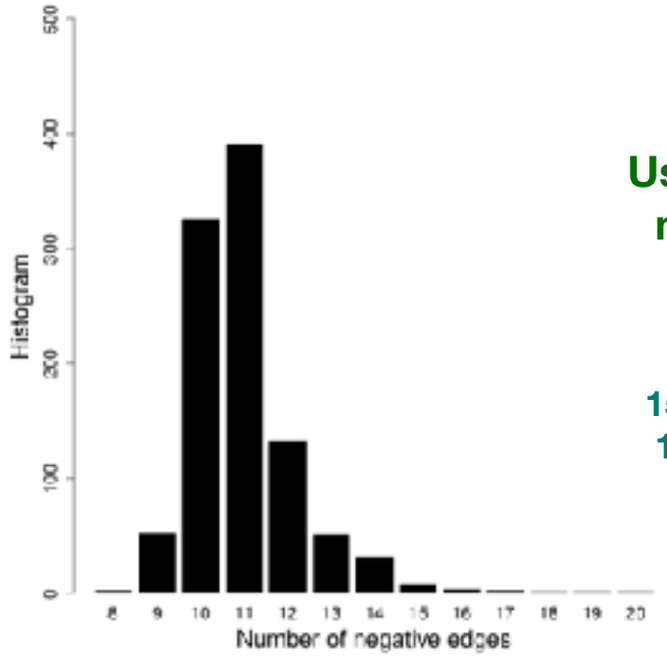
- **Simulation 4**: The same as Simulation 3, but now using the reduced model as the seed.

Results (simulations 1 and 2)

Edge distributions (6 FP)

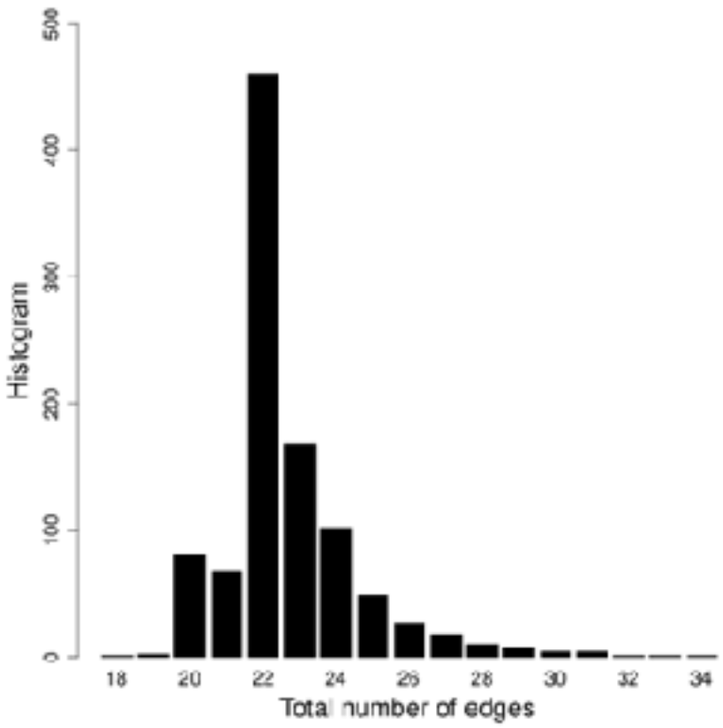


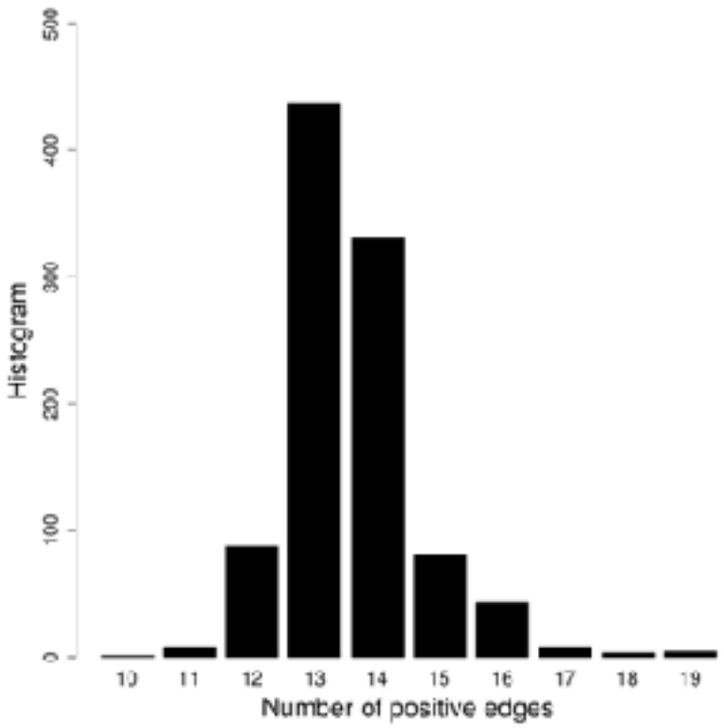


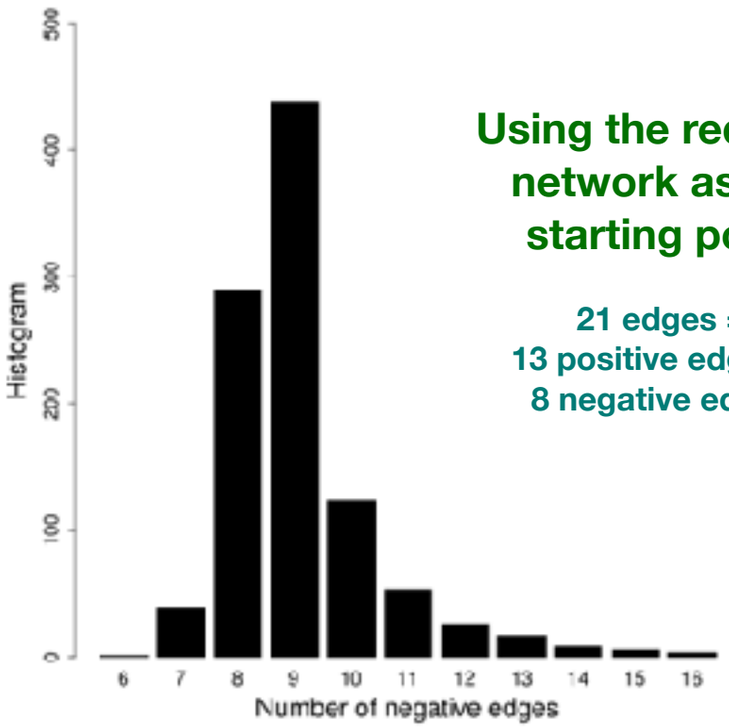


Using the original network as the starting point

25 edges =
15 positive edges +
10 negative edges





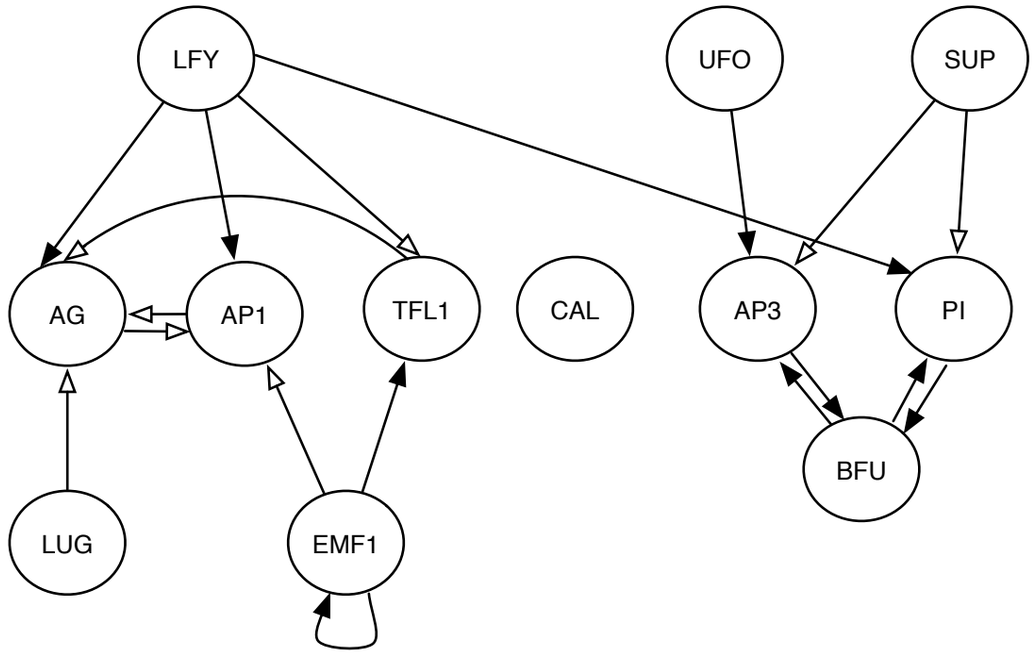


Using the reduced network as the starting point

21 edges =
13 positive edges +
8 negative edges

Synthetic network found with 18 edges (net18)

Contains only the six fixed points of the original network



Using the reduced network as the starting point

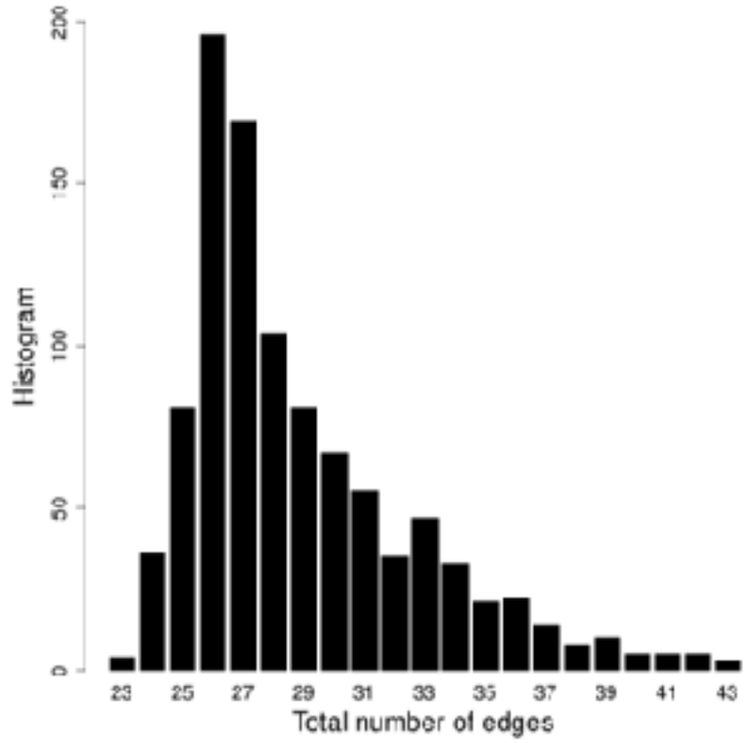
$$W = \begin{array}{c} \\ EMF1 \\ TFL1 \\ LFY \\ AP1 \\ CAL \\ LUG \\ UFO \\ BFU \\ AG \\ AP3 \\ PI \\ SUP \end{array} \begin{array}{c} \begin{array}{cccccccccccc} EMF1 & TFL1 & LFY & AP1 & CAL & LUG & UFO & BFU & AG & AP3 & PI & SUP \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & -2 & 1 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & -2 \\ 0 & 0 & 4 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \quad \Theta = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -2 \\ 1 \\ 0 \\ 4 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

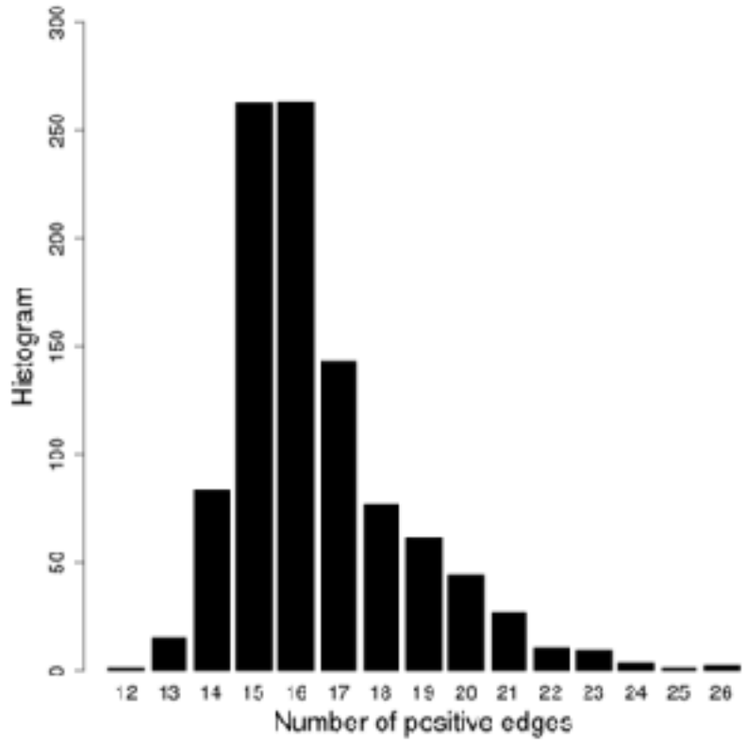
Basin of attraction for the six fixed points

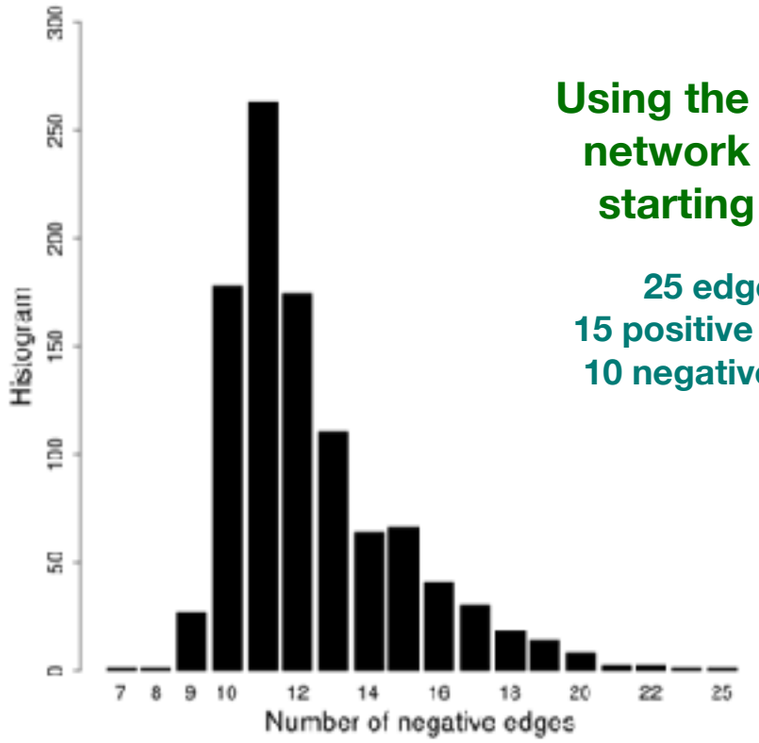
Attractors	Parallel original	Parallel reduced	Parallel net18	BS original	BS reduced	BS net18	Cell types
Fixed point 1	168	168	540	1344	1344	1344	Sep
Fixed point 2	248	248	124	192	192	192	Pet
Fixed point 3	24	24	36	448	448	448	Car
Fixed point 4	8	8	4	64	64	64	Sta
Fixed point 5	384	384	960	1792	1792	1792	Inf
Fixed point 6	384	384	192	256	256	256	Mut

Results (simulations 3 and 4)

Edge distributions (4 FP)

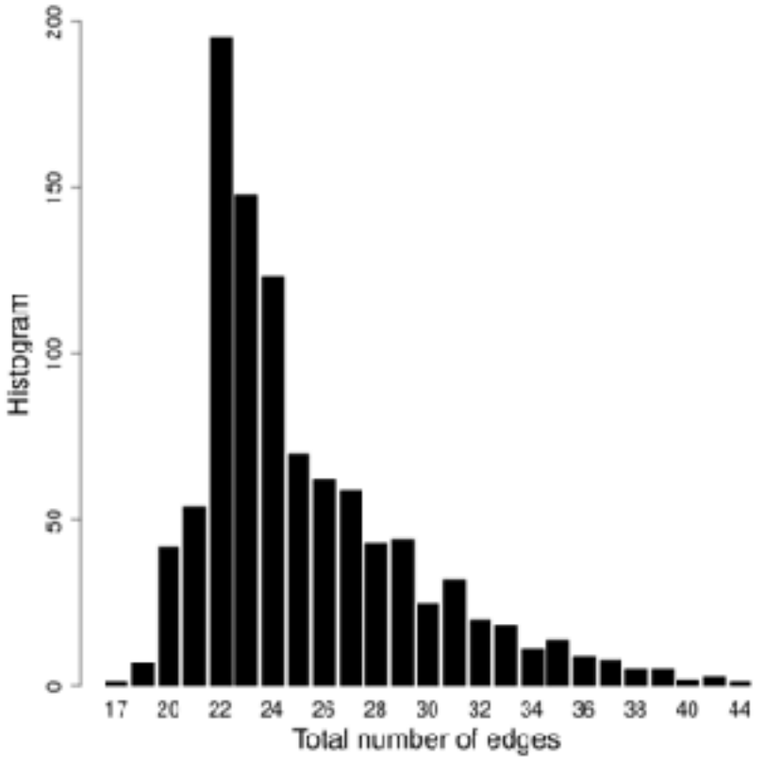


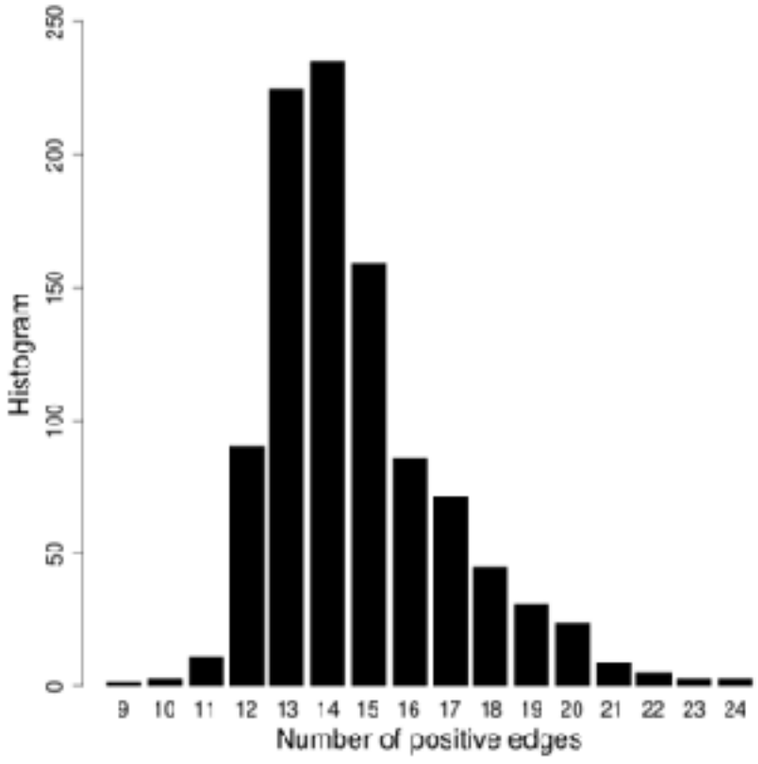


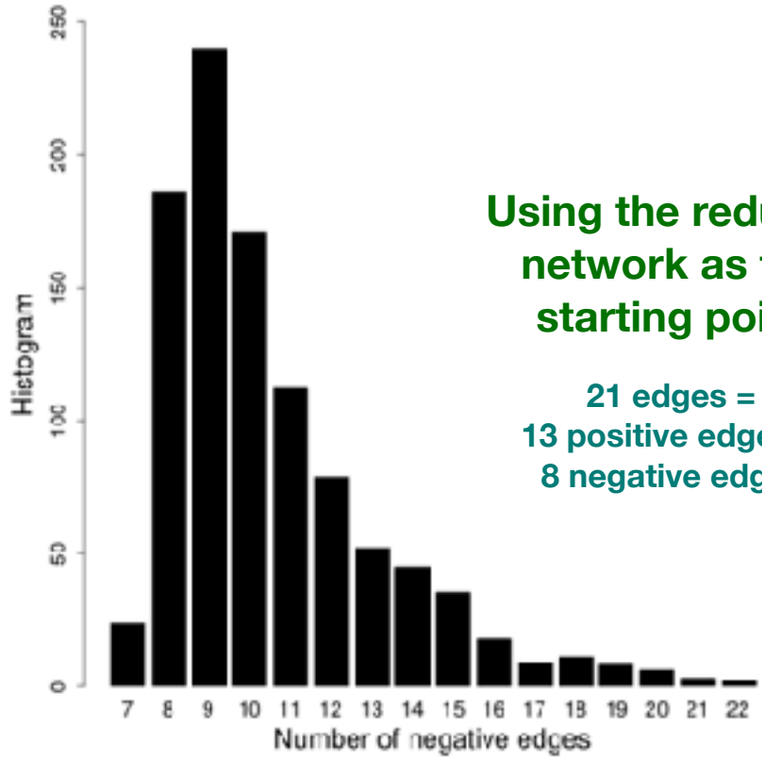


Using the original network as the starting point

25 edges =
15 positive edges +
10 negative edges



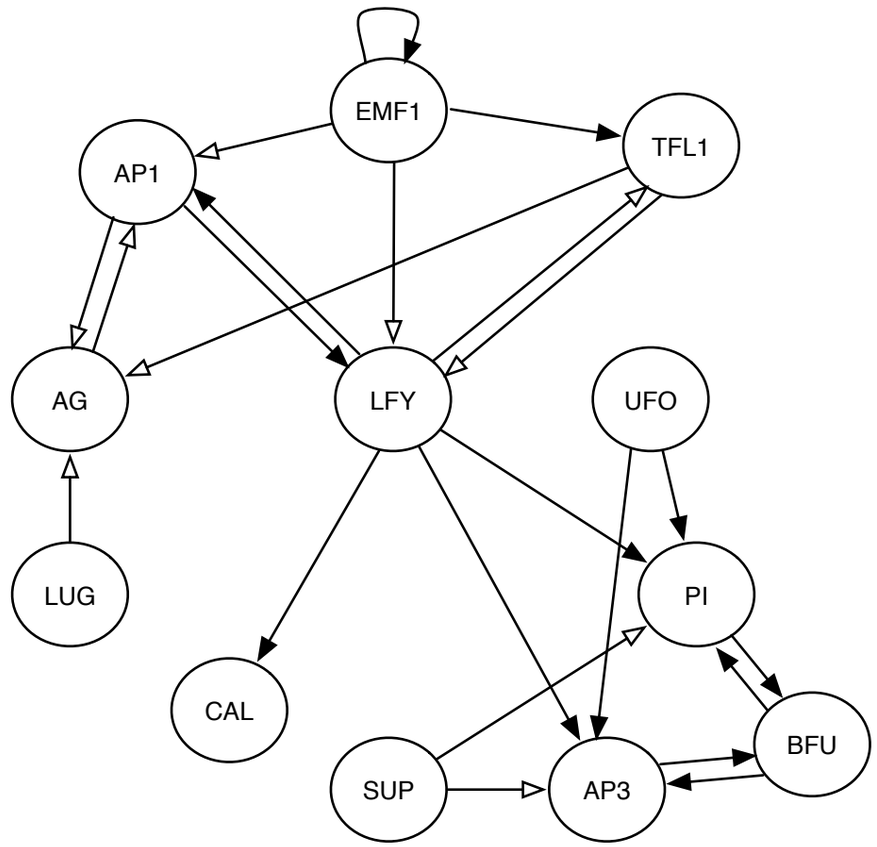




Using the reduced network as the starting point

21 edges =
13 positive edges +
8 negative edges

Synthetic network found with 23 edges (net23)



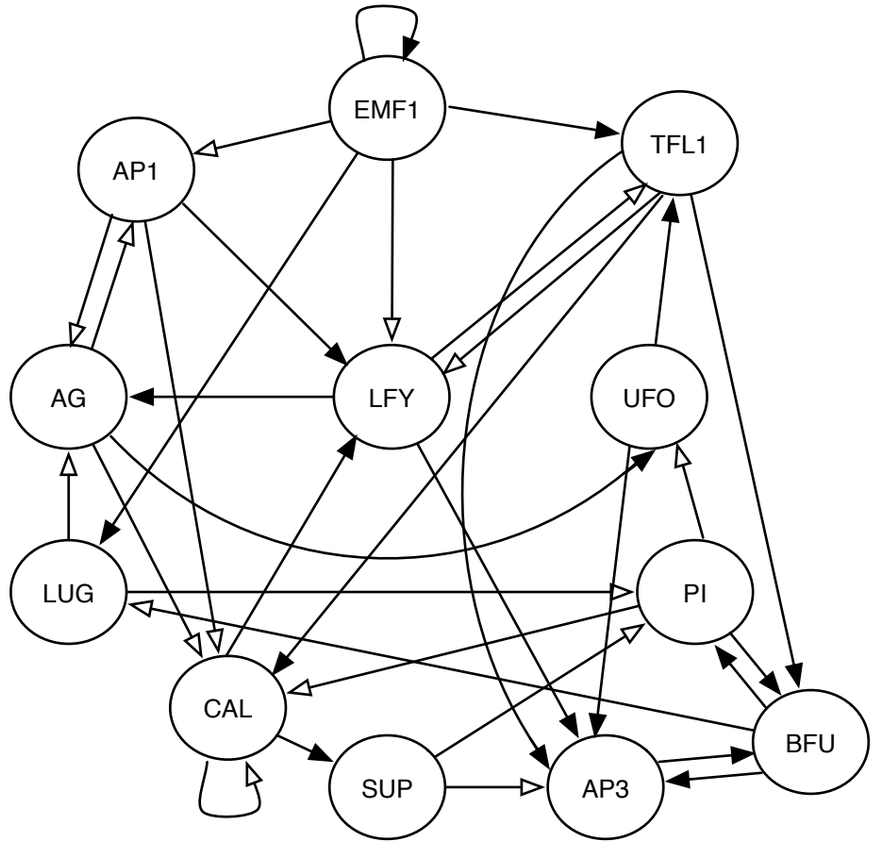
Using the original network as the starting point

Contains only the first four fixed points of the original network

$$W = \begin{pmatrix} & EMF1 & TFL1 & LFY & AP1 & CAL & LUG & UFO & BFU & AG & AP3 & PI & SUP \\ EMF1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ TFL1 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ LFY & -2 & -1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ AP1 & -1 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ CAL & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ LUG & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ UFO & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ BFU & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ AG & 0 & -2 & 0 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ AP3 & 0 & 0 & 3 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & -2 \\ PI & 0 & 0 & 4 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & -1 \\ SUP & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \Theta = \begin{pmatrix} 3 \\ 0 \\ 3 \\ -1 \\ 1 \\ 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Synthetic network found with 34 edges (net34)

Contains only the first four fixed points of the original network, with more evenly distributed basin of attraction per fixed point



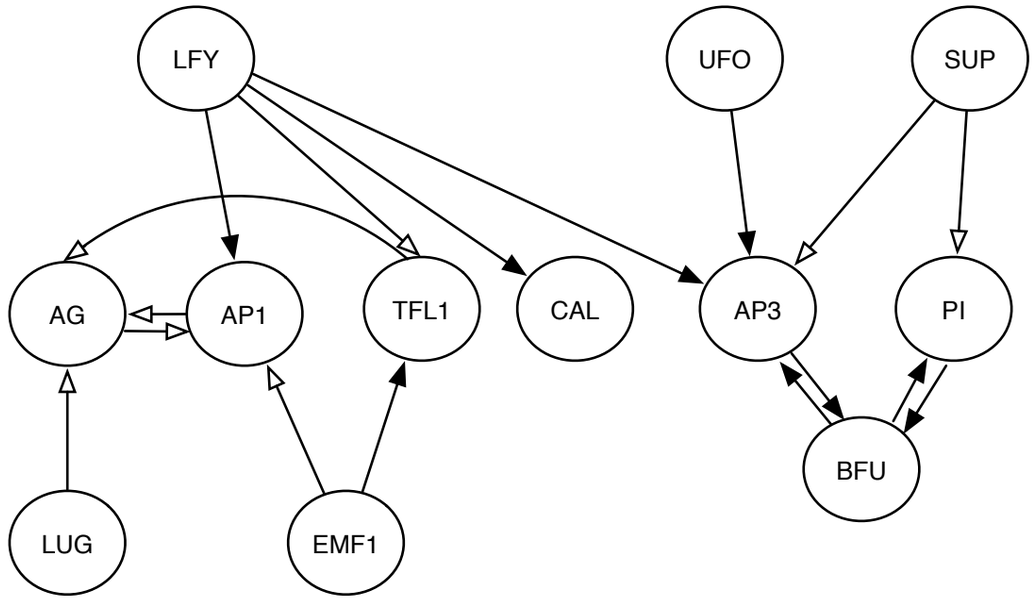
Using the original network as the starting point

$$W = \begin{array}{c} \\ EMF1 \\ TFL1 \\ LFY \\ AP1 \\ CAL \\ LUG \\ UFO \\ BFU \\ AG \\ AP3 \\ PI \\ SUP \end{array} \begin{array}{c} \begin{array}{cccccccccccc} EMF1 & TFL1 & LFY & AP1 & CAL & LUG & UFO & BFU & AG & AP3 & PI & SUP \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -2 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -3 & -4 & 0 & 0 & 0 & -3 & 0 & -3 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 & -5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & -4 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & -4 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 3 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \quad \Theta = \begin{pmatrix} 3 \\ 1 \\ 3 \\ -1 \\ 2 \\ 4 \\ 4 \\ 0 \\ -4 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Basin of attraction for the four fixed points using net23 and net34

Attractors	Parallel net23	BS net23	BS net34	Cell types
Fixed point 1	504	3136	1184	Sep
Fixed point 2	616	448	864	Pet
Fixed point 3	24	448	1184	Car
Fixed point 4	8	64	864	Sta

Synthetic network found with 17 edges (net17)



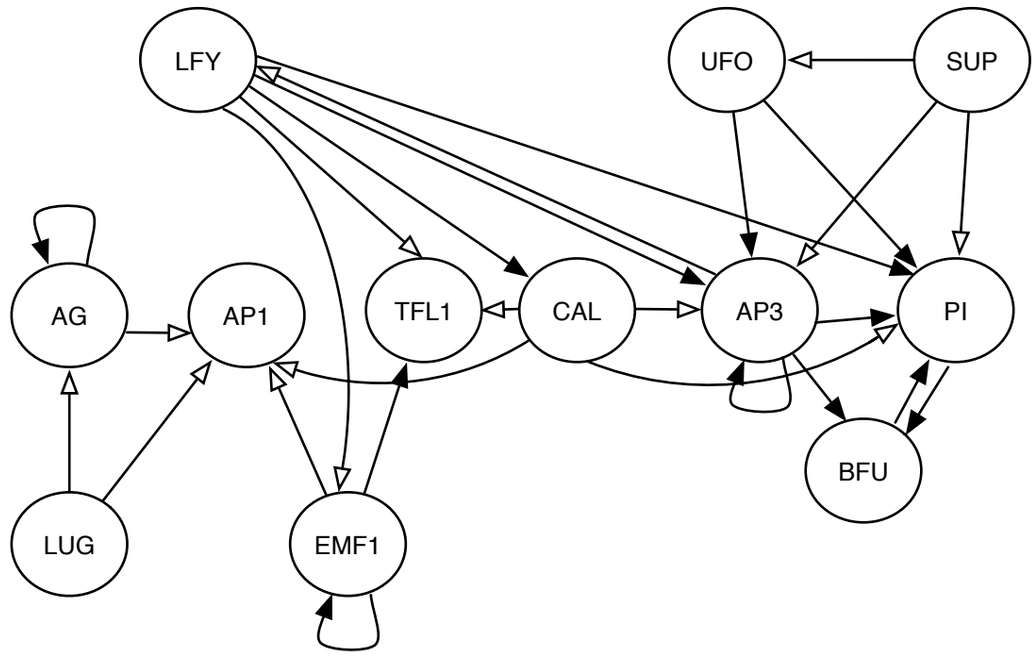
Contains only the first four fixed points of the original network

Using the reduced network as the starting point

$$W = \begin{pmatrix} & EMF1 & TFL1 & LFY & AP1 & CAL & LUG & UFO & BFU & AG & AP3 & PI & SUP \\ EMF1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ TFL1 & 1 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ LFY & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ AP1 & -2 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 \\ CAL & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ LUG & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ UFO & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ BFU & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ AG & 0 & -2 & 0 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ AP3 & 0 & 0 & 3 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & -2 \\ PI & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ SUP & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \Theta = \begin{pmatrix} 0 \\ 5 \\ 0 \\ -2 \\ 1 \\ 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Synthetic network found with 27 edges (net27)

Contains only the first four fixed points of the original network, with evenly distributed basin of attraction per fixed point



Using the reduced network as the starting point

$$W = \begin{pmatrix} & EMF1 & TFL1 & LFY & AP1 & CAL & LUG & UFO & BFU & AG & AP3 & PI & SUP \\ EMF1 & 1 & 0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ TFL1 & 1 & 0 & -2 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ LFY & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ AP1 & -2 & 0 & 0 & 0 & -3 & -1 & 0 & 0 & -2 & 0 & 0 & 0 \\ CAL & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ LUG & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ UFO & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5 \\ BFU & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ AG & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 5 & 0 & 0 & 0 \\ AP3 & 0 & 0 & 3 & 0 & -1 & 0 & 2 & 0 & 0 & 5 & 0 & -2 \\ PI & 0 & 0 & 4 & 0 & -2 & 0 & 1 & 1 & 0 & 3 & 0 & -1 \\ SUP & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \Theta = \begin{pmatrix} 1 \\ 0 \\ 4 \\ -1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 4 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Basin of attraction for the four fixed points using net17 and net27

Attractors	Parallel net17	BS net17	BS net27	Cell types
Fixed point 1	1260	2688	1024	Sep
Fixed point 2	140	384	1024	Pet
Fixed point 3	108	896	1024	Car
Fixed point 4	12	128	1024	Sta

Summary

- An EC approach can effectively reconstruct alternative solutions based on existing gene regulatory models under the threshold Boolean network paradigm.

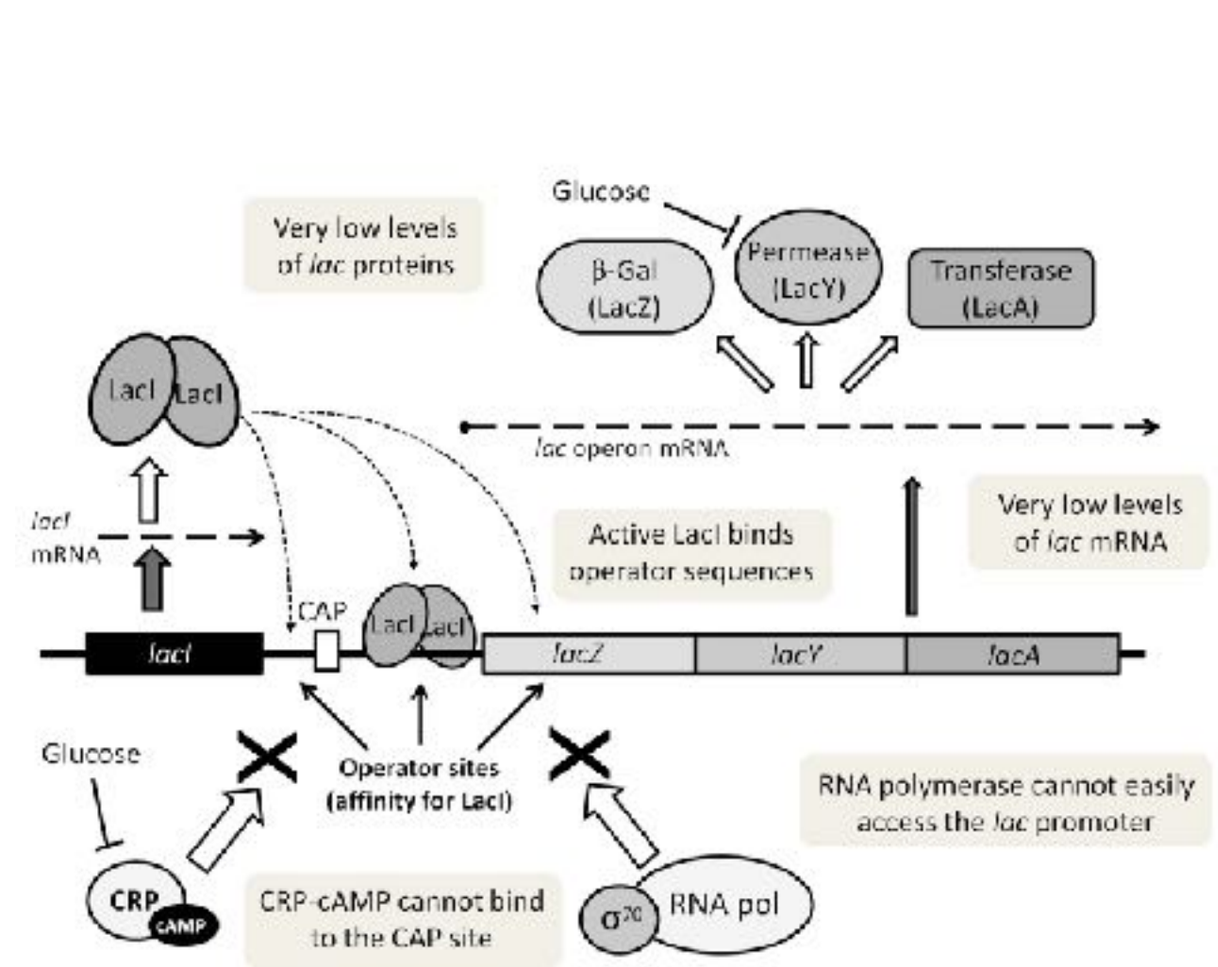
- We were able to find interesting solutions, such as networks with fewer edges than the existing models.

- It was found that in order to change the distribution of the sizes of the basin of attractions, so that each fixed point had more or less the same basin of attraction size, this was achieved by increasing the complexity (number of edges) of the base models used.

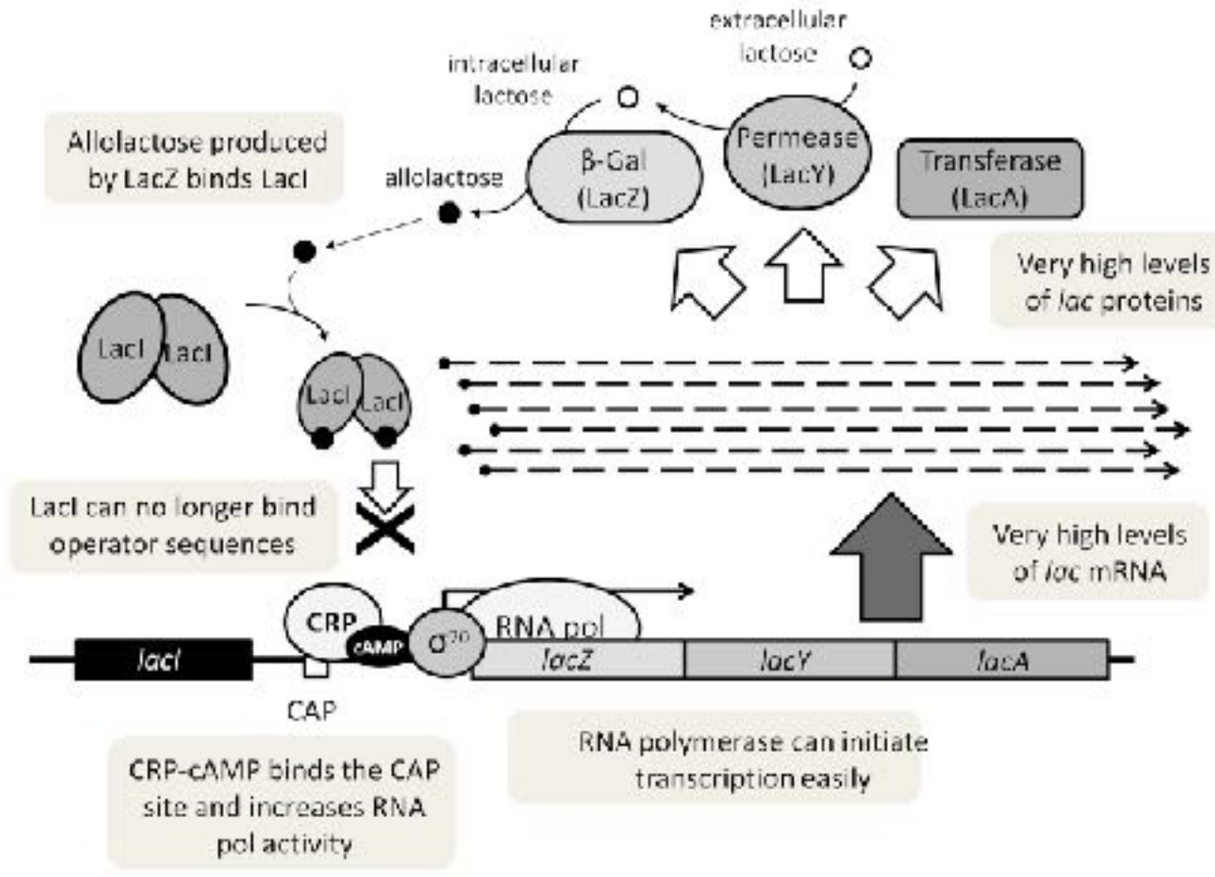
Application 2

Ruz, G.A, Ashlock, D., Ledger, T., Goles, E. Inferring bistable lac operon Boolean regulatory networks using evolutionary computation. The 2017 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB 2017), Manchester, U.K., 23-25 August, 2017, pp. 1-8.

Lac operon in *E. coli*

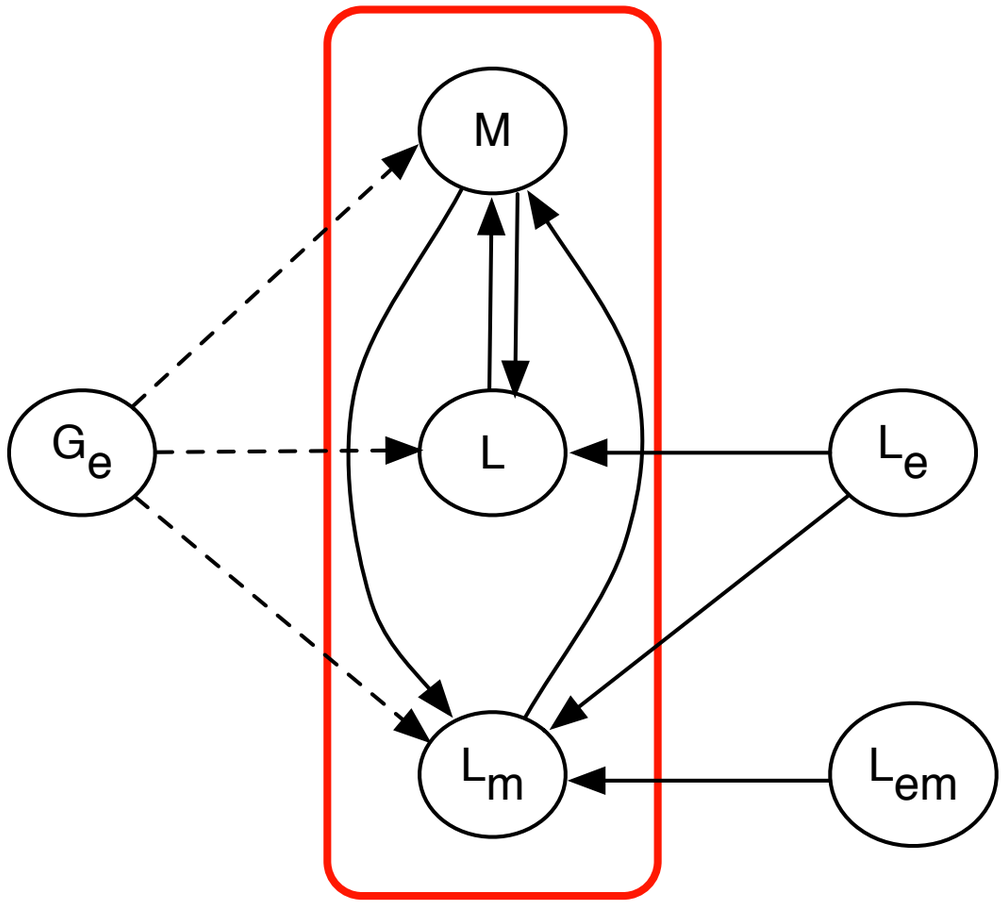


OFF



ON

The reduced *lac* operon Boolean network model



$$
\begin{aligned}
f_M &= \neg G_e \wedge (L \vee L_m) \\
f_L &= M \wedge L_e \wedge \neg G_e \\
f_{L_m} &= ((L_{em} \wedge M) \vee L_e) \wedge \neg G_e
\end{aligned}
$$

A. Veliz-Cuba, B. Stigler. Journal of Computational Biology 18 (6) (2011) 783-794.

The dynamics

The dynamics

- **Case 1:** $G_e = 1$. All configurations eventually reach the unique steady state $F_1 = (0, 0, 0)$ (operon being OFF).

The dynamics

- **Case 1:** $G_e = 1$. All configurations eventually reach the unique steady state $F_1 = (0, 0, 0)$ (operon being OFF).

- **Case 2:** $G_e = L_e = L_{em} = 0$. All configurations eventually reach the unique steady state $F_1 = (0, 0, 0)$ (operon being OFF).

The dynamics

- **Case 1:** $G_e = 1$. All configurations eventually reach the unique steady state $F_1 = (0, 0, 0)$ (operon being OFF).

- **Case 2:** $G_e = L_e = L_{em} = 0$. All configurations eventually reach the unique steady state $F_1 = (0, 0, 0)$ (operon being OFF).

- **Case 3:** $G_e = L_e = 0$ and $L_{em} = 1$. All configurations eventually reach one of the two steady states; $F_1 = (0,0,0)$ or $F_2 = (1, 0, 1)$ (operon being OFF and ON, respectively). That is, the model is bistable.

The dynamics

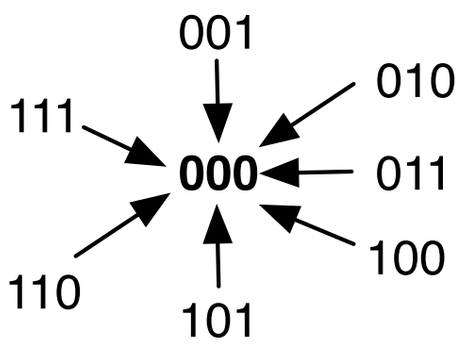
- **Case 1:** $G_e = 1$. All configurations eventually reach the unique steady state $F_1 = (0, 0, 0)$ (operon being OFF).

- **Case 2:** $G_e = L_e = L_{em} = 0$. All configurations eventually reach the unique steady state $F_1 = (0, 0, 0)$ (operon being OFF).

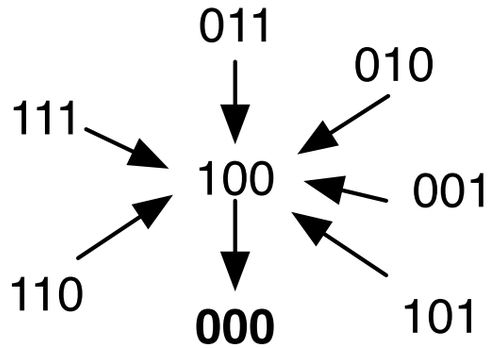
- **Case 3:** $G_e = L_e = 0$ and $L_{em} = 1$. All configurations eventually reach one of the two steady states; $F_1 = (0,0,0)$ or $F_2 = (1, 0, 1)$ (operon being OFF and ON, respectively). That is, the model is bistable.

- **Case 4:** $G_e = 0$ and $L_e = L_{em} = 1$. All configurations eventually reach the unique steady state $F_3 = (1, 1, 1)$ (operon being ON).

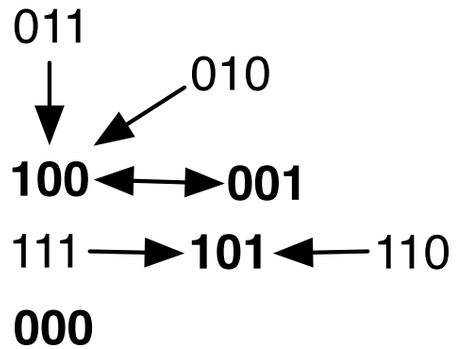
Sate transition graphs of the reduced *lac* operon network



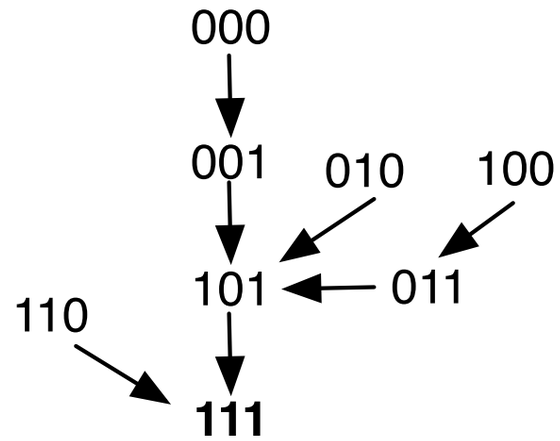
case 1



case 2



case 3



case 4

Problem description

- We propose to use evolutionary computation (EC) to infer bistable threshold Boolean regulatory networks, applied to the reconstruction of the reduced *lac* operon model.

- EC to be considered: differential evolution (DE), genetic algorithm (GA), and particle swarm optimization (PSO).

- We will measure: the error of the inferred networks or effectiveness, the average time to find a solution, the average number of iterations, and topological characteristics of the networks found, for each approach.

Fitness function

- The proposed fitness function is the the *total hit rate error e* of the network, defined by

$$e = e_1 + e_2 + e_3 + e_4$$

- where e_i corresponds to the hit rate error of the network for case i (for $i = 1 \ldots 4$).

Fitness function

- To compute e_1 for a given candidate solution (network), we used the following procedure:

1) Starting from each configuration of the network, update the network $k = 2^n$ times, and record its final configuration.

2) Count how many of the final configurations equal to F_1. Let m_1 be the total counts.

3) Then e_1 is computed by

$$e_1 = 1 - \frac{m_1}{nk}$$

- For e_2 the same procedure is used, but considering the configuration of the control parameters of case 2.

Fitness function

- To compute e_3 we used the following procedure:

1) Starting from each configuration of the network, update the network $k = 2^n$ times, and record its final configuration.

2) Count how many of the final configurations equal to F_1, call this value m_{3a}. Count how many of the final configurations equal to F_2, call this value m_{3b}.

3) If $m_{3a} = nk$ or $m_{3b} = nk$ then $e_3 = 0.5$, else

$$e_3 = 1 - \frac{(m_{3a} + m_{3b})}{nk}$$

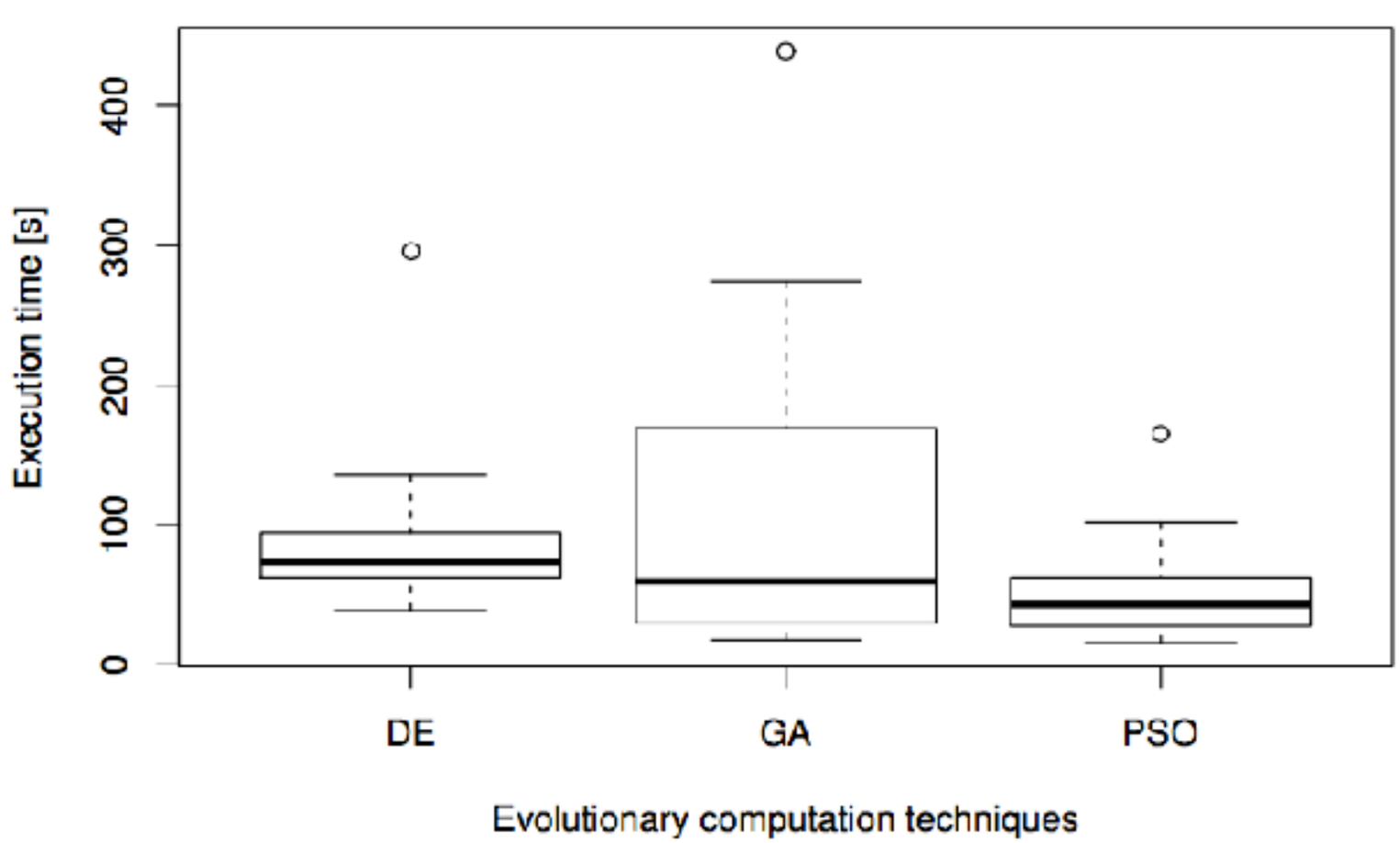
- For e_4 we used the same procedure as e_1, but considering the configuration of the control parameters of case 4 and in 2) we considered the counts that equal to F_3 (thus, obtaining m_4).

Methods

- So, it's a minimization problem, where a correct solution is obtained when $e = 0$.
- The simulations were carried out using the open-source R software environment for statistical computing running on a 2.8 GHz Intel Core i7 and 8 GB-RAM computer.
- For DE, we used the function *DEoptim,* for GA we use the function *ga*, and for PSO we used *psoptim.*
- For each method we used a population of size 50, maximum iteration of 500, and the search range for the network values (weights and thresholds) was set to the real interval [-5;5].
- Default values were considered for the user defined parameters for DE, GA, and PSO.
- We ran each algorithm 30 times.

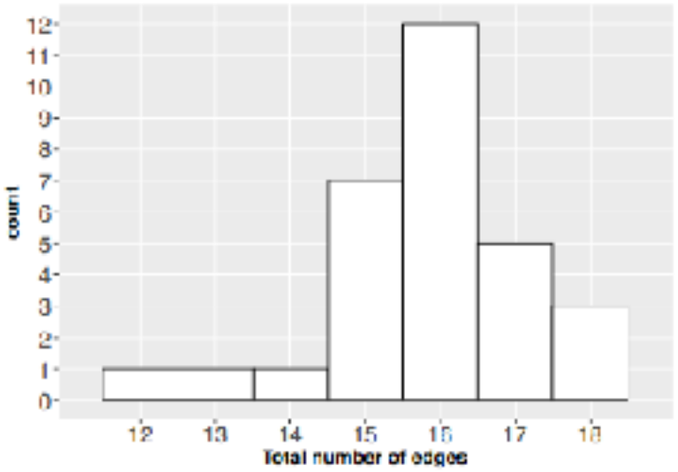
Results: comparisons between DE, GA, and PSO

Evo. Comp. Technique	Effectiveness	Avg.±SD. execution time [s]	Avg.±SD. number of iterations
DE	100%	84.6±46.5	112.3±35.1
GA	67%	109.9±111.8	191.9±196.1
PSO	90%	51.5±33.2	78.9±50.9



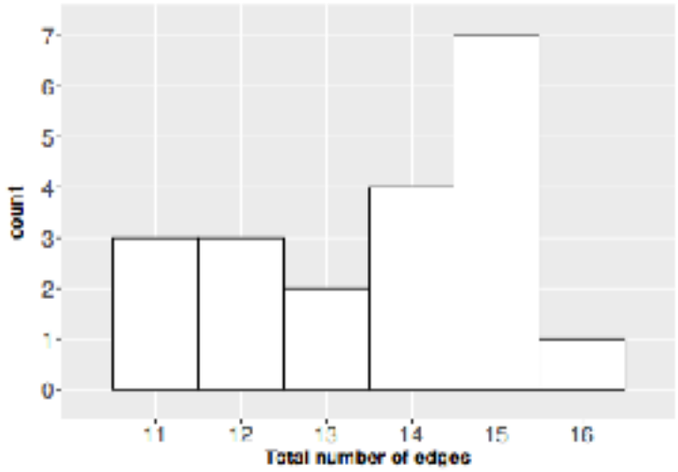
Results: topology distributions of the inferred networks

DE



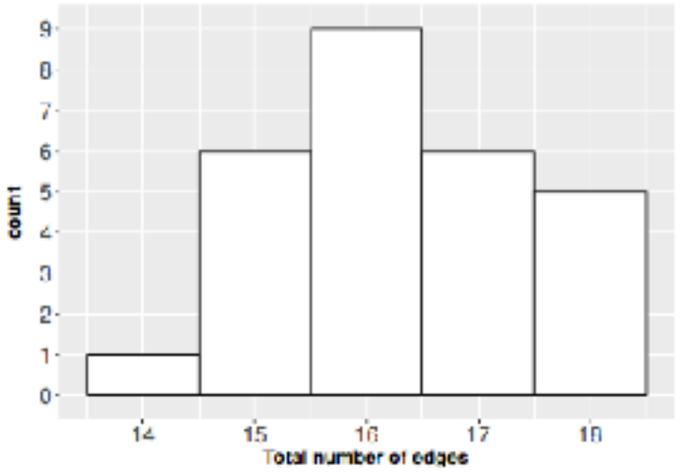
(a)

GA

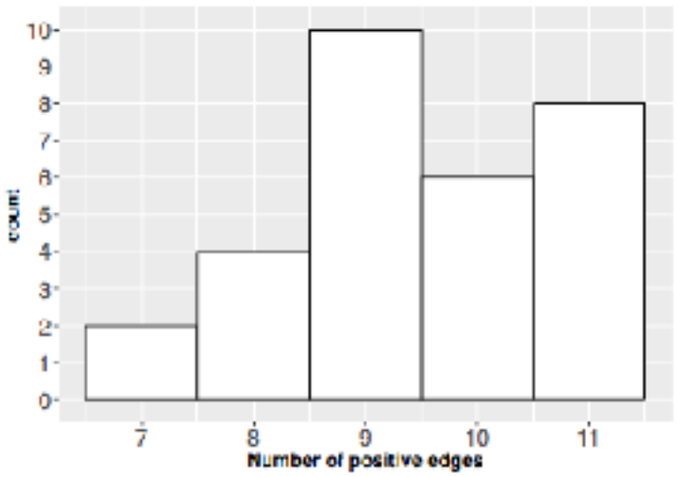


(b)

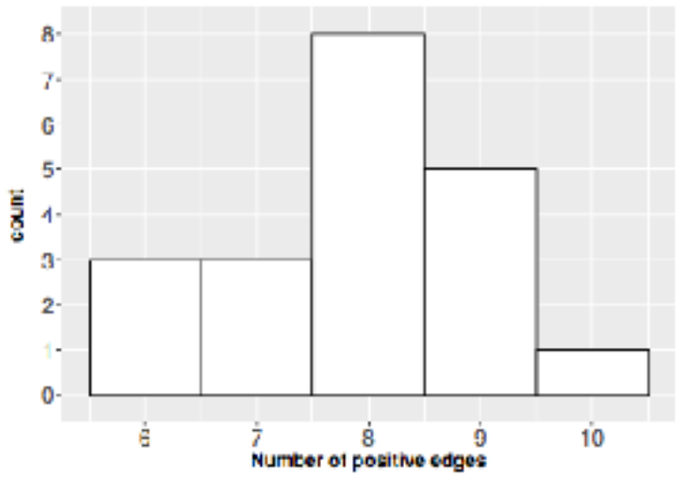
PSO



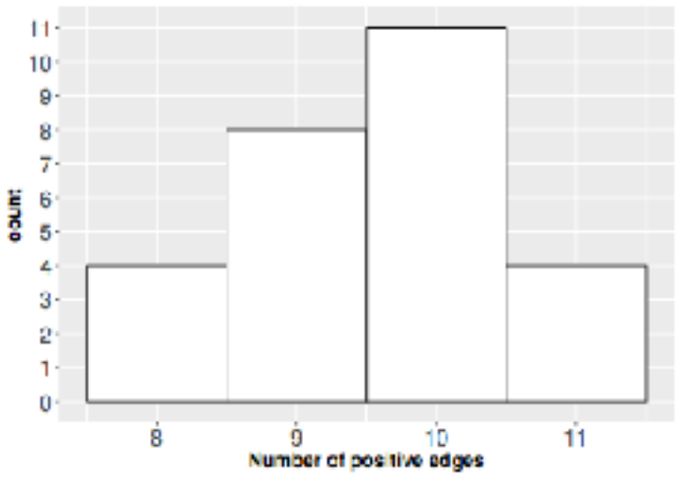
(c)



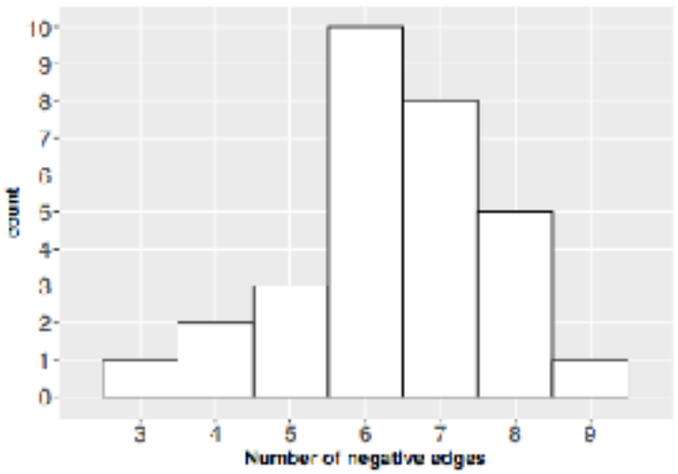
(d)



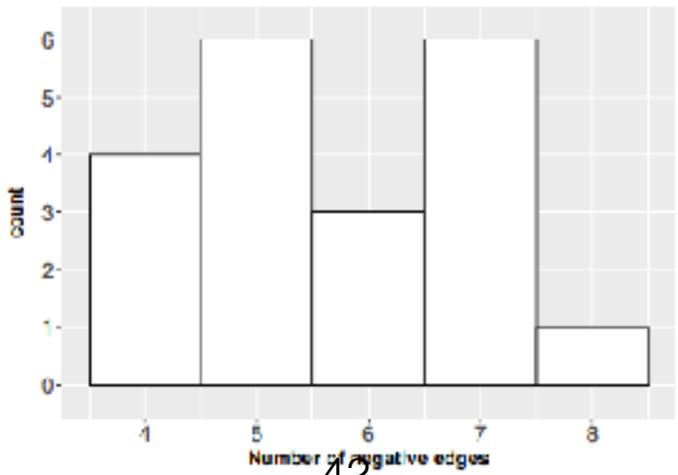
(e)



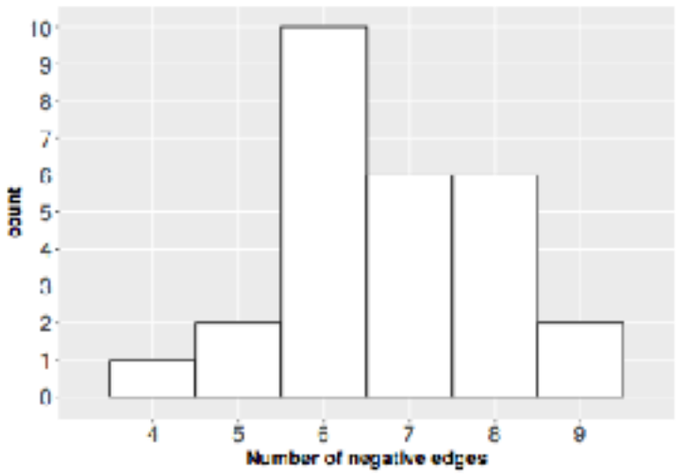
(f)



(g)

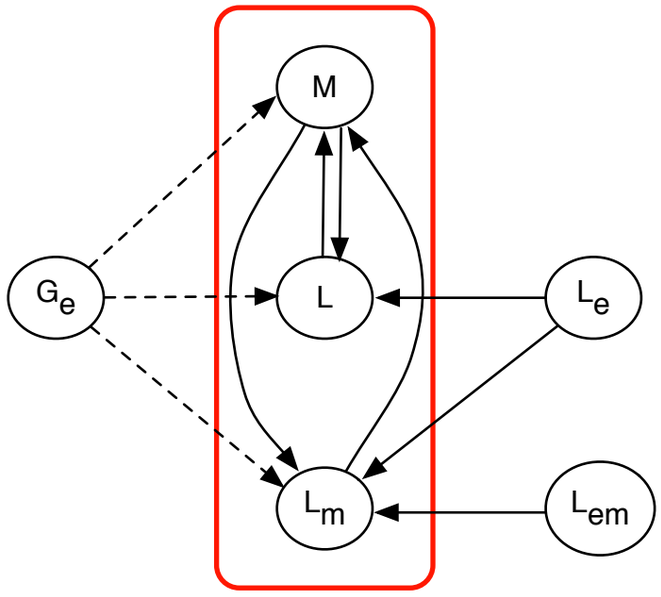


(h)



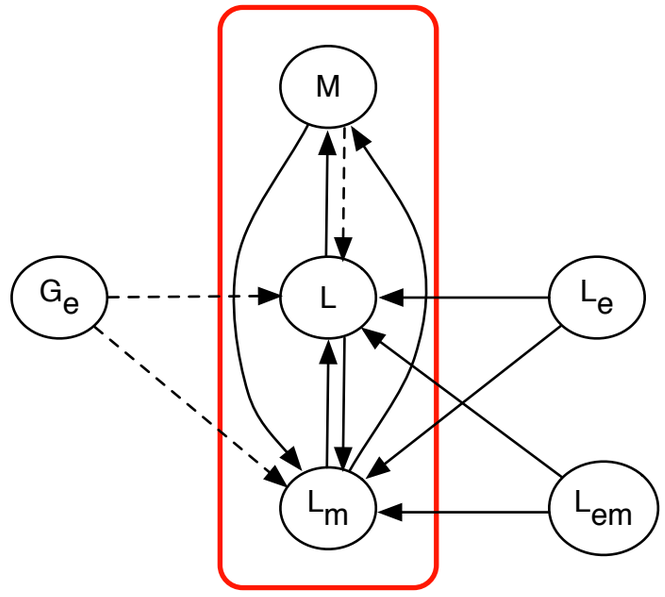
(i)

Example of an inferred network



$$
\begin{aligned}
f_M &= \neg G_e \wedge (L \vee L_m) \\
f_L &= M \wedge L_e \wedge \neg G_e \\
f_{L_m} &= ((L_{em} \wedge M) \vee L_e) \wedge \neg G_e
\end{aligned}
$$

Original network

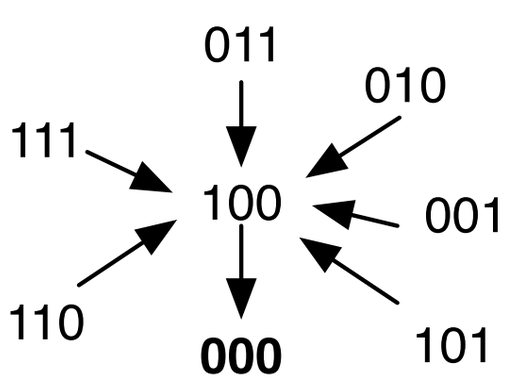


$$
W = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 2 & 3 \\
-3 & 4 & 3 & -5 & 0 & 3 \\
-3 & 2 & 1 & 2 & 3 & 0
\end{bmatrix}
$$

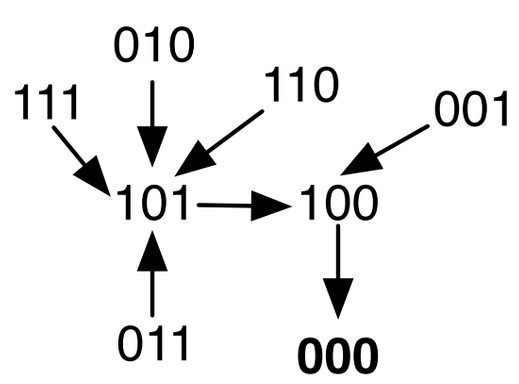
$$
\Theta = \begin{bmatrix} 0 & 0 & 0 & 0.6 & 4.1 & 2.3 \end{bmatrix}
$$

Inferred network using DE

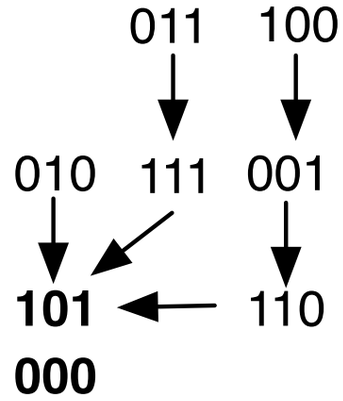
Sate transition graphs for the four cases of the inferred network



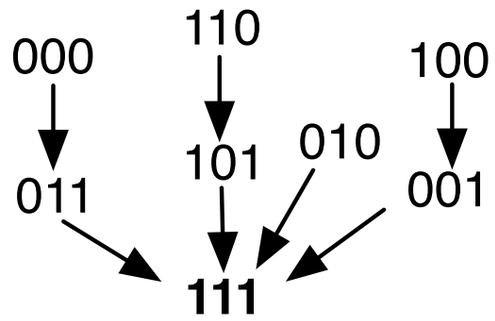
case 1



case 2



case 3



case 4

Summary

Summary

- Simulations showed that the three evolutionary computation techniques were capable of finding solutions being DE the most effective.

Summary

- Simulations showed that the three evolutionary computation techniques were capable of finding solutions being DE the most effective.
- Solutions were found only defining very general parameters like the population size, the maximum iterations, and the range of the values to look for.

Summary

- Simulations showed that the three evolutionary computation techniques were capable of finding solutions being DE the most effective.
- Solutions were found only defining very general parameters like the population size, the maximum iterations, and the range of the values to look for.
- We reported as an example, one of the networks inferred by DE, which presented biologically meaningful characteristics.

Summary

- Simulations showed that the three evolutionary computation techniques were capable of finding solutions being DE the most effective.

- Solutions were found only defining very general parameters like the population size, the maximum iterations, and the range of the values to look for.

- We reported as an example, one of the networks inferred by DE, which presented biologically meaningful characteristics.

- Given the way the fitness function is constructed, no spurious limit cycles can appear, in contrast to the original model.

Summary

- Simulations showed that the three evolutionary computation techniques were capable of finding solutions being DE the most effective.

- Solutions were found only defining very general parameters like the population size, the maximum iterations, and the range of the values to look for.

- We reported as an example, one of the networks inferred by DE, which presented biologically meaningful characteristics.

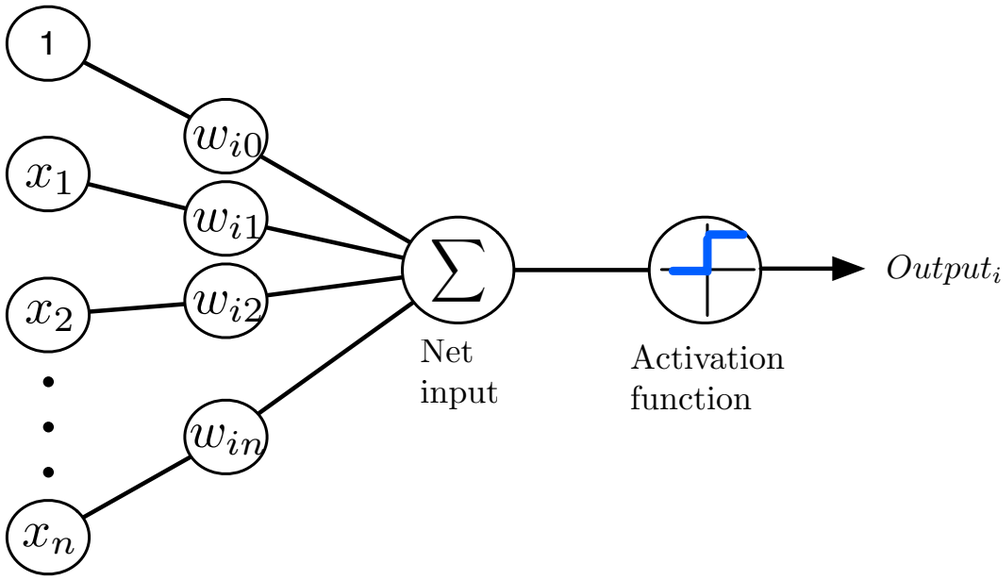
- Given the way the fitness function is constructed, no spurious limit cycles can appear, in contrast to the original model.

- Future results will consider exploring different parameter values of the algorithms in order to reduce the variability (specially in GA) in the runtime and number of iterations. Also we will consider working with the full ten node network *lac* operon model.

Application 3

Ruz, G.A., Zúñiga, A., Goles, E. A Boolean network model of bacterial quorum-sensing systems, *International Journal of Data Mining and Bioinformatics*, Vol. 21, 2018, 123-144.

Neural network approach for regulatory network reconstruction



$$x_i(t+1) = H\left(\sum_{j=0}^{n} \omega_{ij} x_j(t)\right) \qquad (1)$$

The learning algorithm for the perceptron is as follows

1 Initialise all the weights to zero (or a small random number)

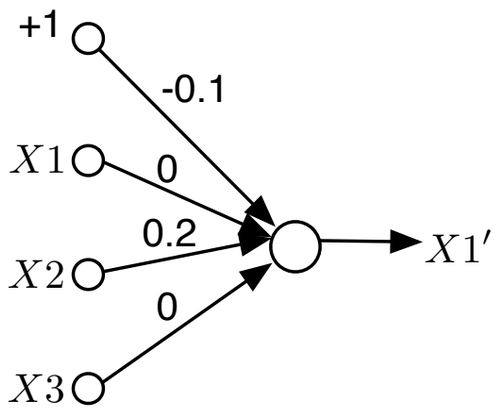
2 For each *k*-th data point (training sample)

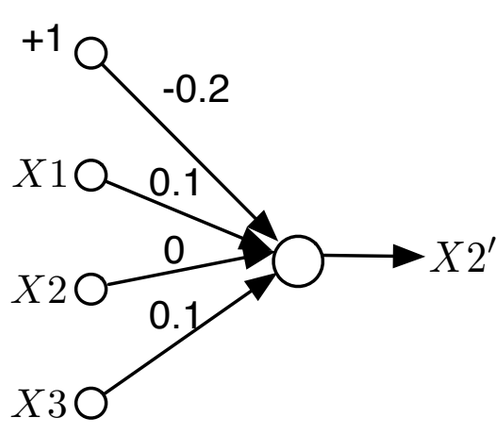
(a) Calculate the output value using (1)

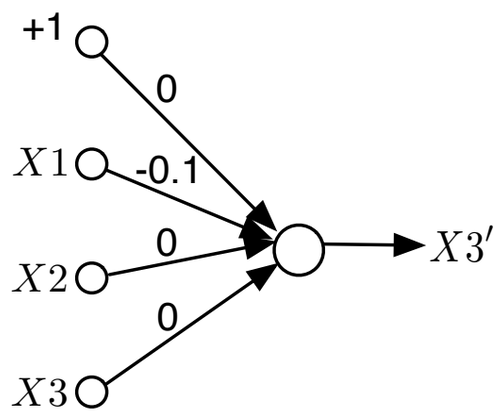
(b) Update the weights using the following rule

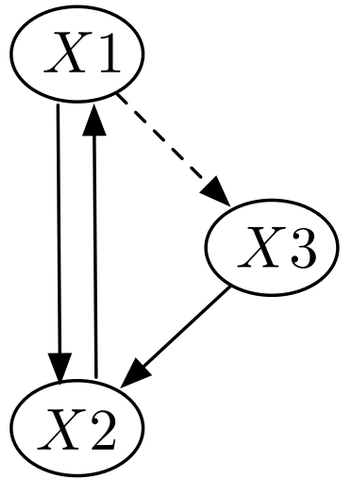
$$w_j = w_j + \tau\left(Target^k - Output^k\right)x_j^k$$

Perceptrons trained with the data of the toy model (slide 4) and the resulting threshold Boolean network model









$$W = \begin{bmatrix} 0 & 0.2 & 0 \\ 0.1 & 0 & 0.1 \\ -0.1 & 0 & 0 \end{bmatrix}$$

$$\Theta = \begin{bmatrix} 0.1 & 0.2 & 0 \end{bmatrix}$$

Problem description

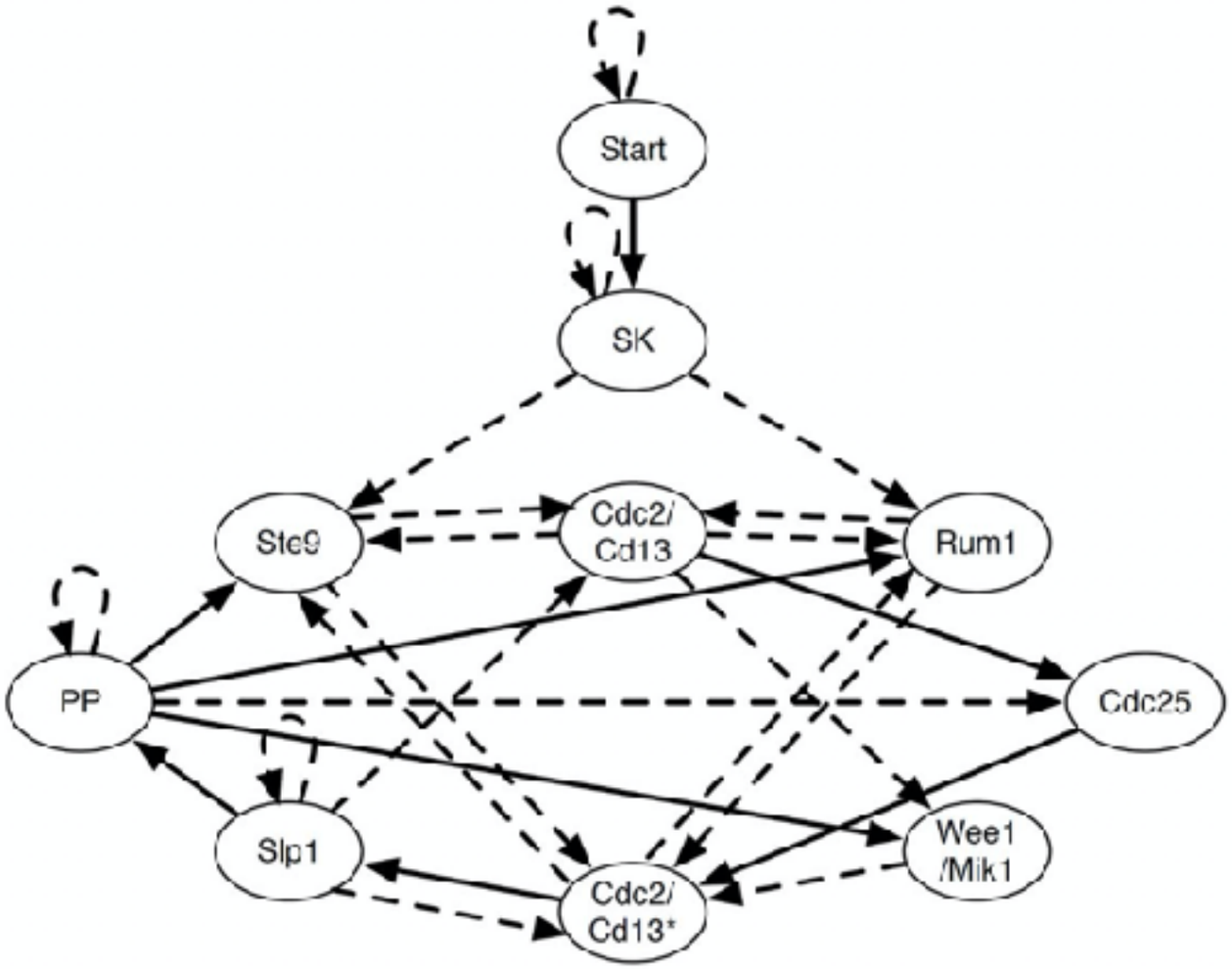
- We will apply this neural network approach first to reconstruct the fission yeast cell cycle network (Davidich and Bornholdt, 2008), and then to to reconstruct a regulatory network model of QS systems in P. phytofirmans PsJN.

- The results will be compared with the networks inferred by an information theoretic approach called REVEAL (Liang et al., 1998), which computes mutual information between genes to identify relations between them,

- and the Best-Fit extension algorithm (Lahdesmaki et al., 2003), which performs an exhaustive search for regulatory genes considering all the possible combinations for a target gene and finds the best solution (function) that minimises the errors.

- The implementations in the R package BoolNet (Müssel et al., 2010) will be considered.

The fission yeast cell cycle network



$$x_i(t+1) = u\left(\sum_{j=1}^{n} w_{ij}x_j - \theta_i\right)$$

$$= \begin{cases} 0, & if \ \sum_{j=1}^{n} w_{ij}x_j - \theta_i < 0 \\ 1, & if \ \sum_{j=1}^{n} w_{ij}x_j - \theta_i > 0 \\ x_i(t), & if \ \sum_{j=1}^{n} w_{ij}x_j - \theta_i = 0 \end{cases}$$

Temporal evolution of sate vectors defining the fission yeast cell cycle

Time	*Start*	*SK*	*Cdc2/ Cdc13*	*Ste9*	*Rum1*	*Slp1*	*Cdc2/ Cd13**	*Wee1/ Mik1*	*Cdc25*	*PP*	*Phase*
1	1	0	0	1	1	0	0	1	0	0	START
2	0	1	0	1	1	0	0	1	0	0	G_1
3	0	0	0	0	0	0	0	1	0	0	G_1/S
4	0	0	1	0	0	0	0	1	0	0	G_2
5	0	0	1	0	0	0	0	0	1	0	G_2
6	0	0	1	0	0	0	1	0	1	0	G_2/M
7	0	0	1	0	0	1	1	0	1	0	G_2/M
8	0	0	0	0	0	1	0	0	1	1	M
9	0	0	0	1	1	0	0	1	0	1	M
10	0	0	0	1	1	0	0	1	0	0	G_1

Topology evaluation

- **Precision** is defined as the fraction of correctly inferred connections out of all the predictions

- **Recall** is the fraction of inferred connections among the true connections in YeastNet (original network)

- **Topology accuracy** is the fraction of correct predictions

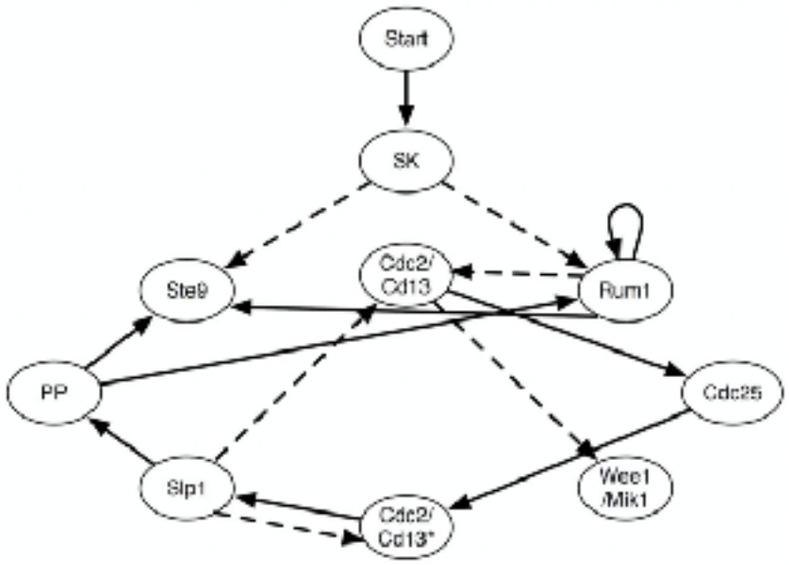
- Also, precision and recall can be combined in a single measure known as the **F-score** (harmonic mean of the precision and recall)

The dynamics accuracy

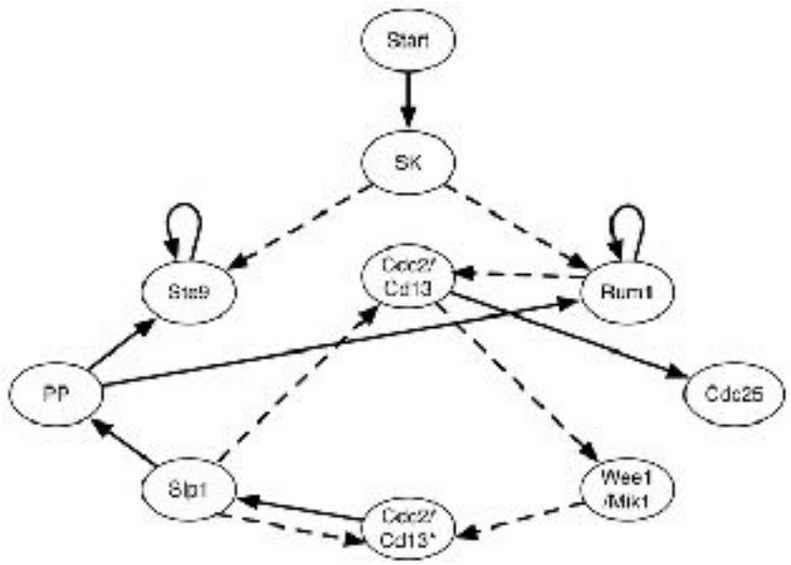
- The dynamics accuracy (the fraction of correct predictions at the bit level) can be computed by

$$Dynamics\ accuracy = \frac{1}{1024 \cdot 10} \sum_{i=0}^{1023} \sum_{j=1}^{10} I\left(YeastNet(i,j), InfNet(i,j)\right)$$

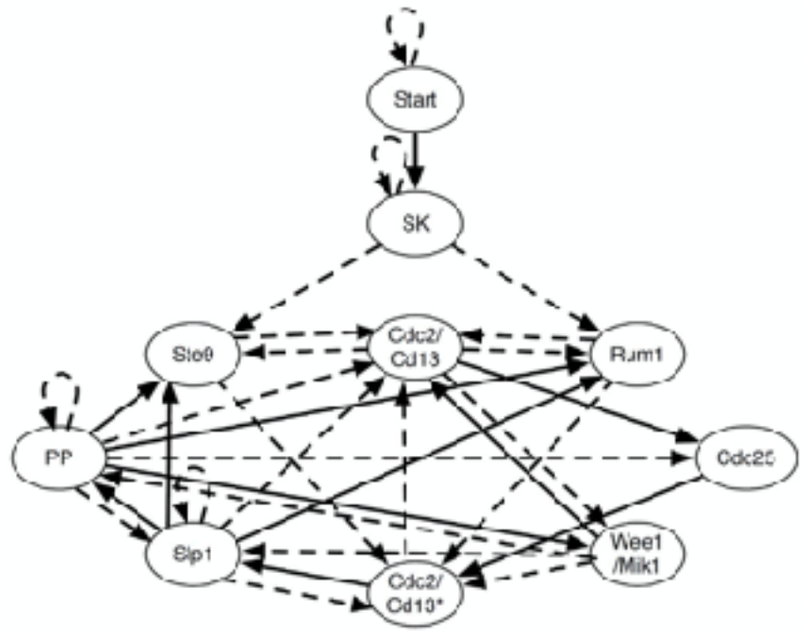
Results



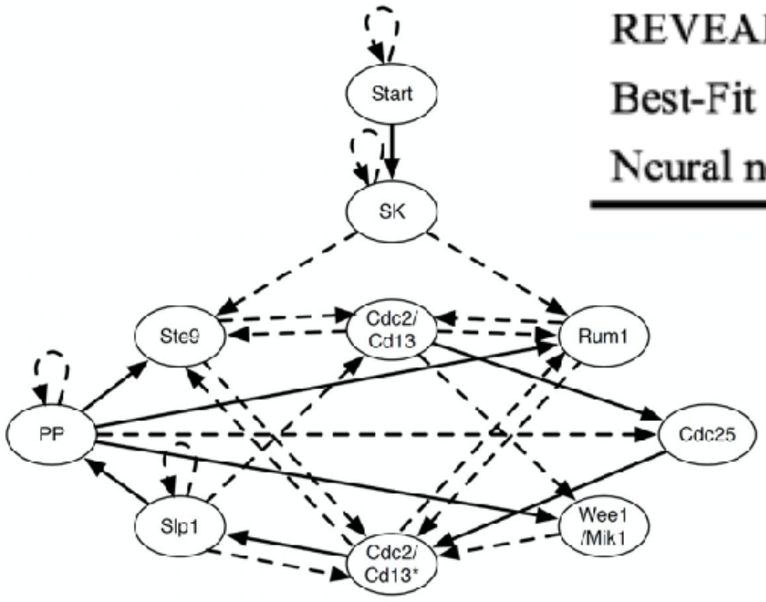
REVEAL



Best-Fit



Neural network



Original

Method	*Precision*	*Recall*	*F-score*	*Topology accuracy*	*Dynamics accuracy*
REVEAL	0.87	0.48	0.62	0.84	0.86
Best-Fit	0.77	0.48	0.59	0.82	0.87
Neural networks	0.76	0.93	0.84	0.90	0.85

Results (II)

- The original model YeastNet has 13 attractors (12 fixed points and 1 limit cycle of length 3). Of particular interest is the fixed point shown in the Table at t = 10 which corresponds to the value 100, in decimal representation, and that represents the G1 phase in the cell cycle with the largest basin of attraction of 762 configurations.

- For the network inferred by REVEAL, all the configurations converge to fixed point 100.

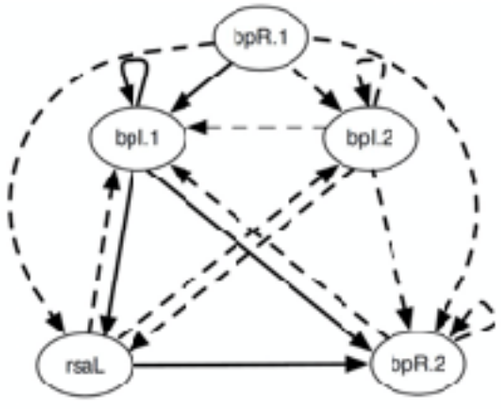
- When using Best-Fit, the configurations converged to two fixed points: 1022 configurations converged to fixed point 100, and the remaining 2 states converged to fixed point 36 (also present in YeastNet).

- Using the neural network approach, the resulting network has eight fixed points, with fixed point 100 having a basin of attraction of 977. The rest of the configurations converge to the remaining seven fixed points, five of which are present in YeastNet (fixed points: 36, 38, 68, 70, and 102), and two that are not (fixed points: 4 and 6). None of the three inferred network presented limit cycles.

Temporal evolution of sate vectors used to reconstruct a regulatory network of bacterial quorum-sensing systems

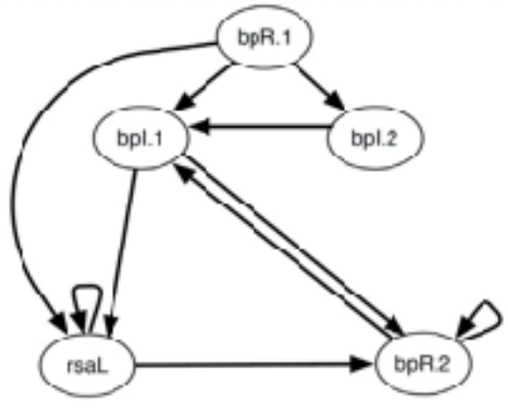
Time	*bpI.1*	*bpI.2*	*bpR.1*	*bpR.2*	*rsaL*
1	0	0	0	0	0
2	0	1	1	0	1
3	0	0	1	0	0
4	1	0	1	0	0
5	1	0	1	0	1
6	1	0	1	1	1
7	0	0	1	0	1

Reconstruction of QS networks

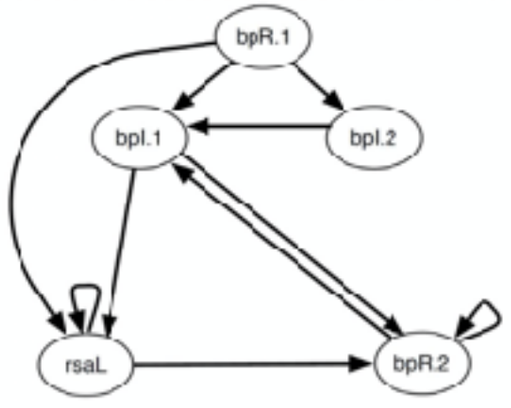


$$W = \begin{bmatrix} 0.2 & -0.2 & 0.3 & -0.4 & -0.1 \\ 0 & -0.1 & -0.1 & 0 & -0.1 \\ 0 & 0 & 0 & 0 & 0 \\ 0.1 & -0.2 & -0.1 & -0.4 & 0.3 \\ 0.3 & -0.1 & -0.1 & 0 & 0 \end{bmatrix}$$

$$\Theta = \begin{bmatrix} 0.1 & 0 & 0 & 0.3 & 0 \end{bmatrix}$$

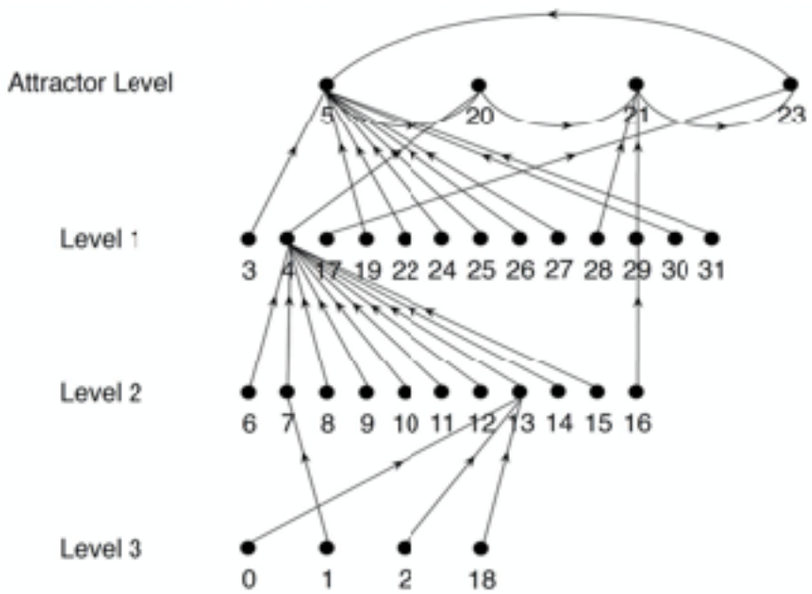


$$\begin{aligned} bpI.1 &= \neg bpI.2 \wedge bpR.1 \wedge \neg bpR.2 \\ bpI.2 &= \neg bpR.1 \\ bpR.1 &= 1 \\ bpR.2 &= bpI.1 \wedge \neg bpR.2 \wedge rsaL \\ rsaL &= (\neg bpI.1 \wedge \neg bpR.1 \wedge \neg rsaL) \vee (bpI.1 \wedge bpR.1) \end{aligned}$$

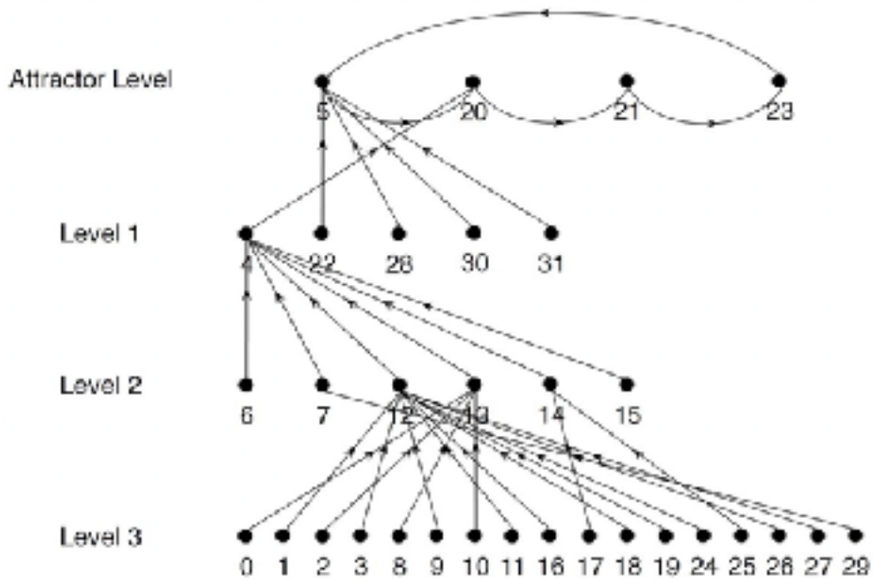


$$\begin{aligned} bpI.1 &= \neg bpI.2 \wedge bpR.1 \wedge \neg bpR.2 \\ bpI.2 &= \neg bpR.1 \\ bpR.1 &= 1 \\ bpR.2 &= bpI.1 \wedge \neg bpR.2 \wedge rsaL \\ rsaL &= (\neg bpI.1 \wedge \neg bpR.1 \wedge \neg rsaL) \vee (bpI.1 \wedge bpR.1) \end{aligned}$$

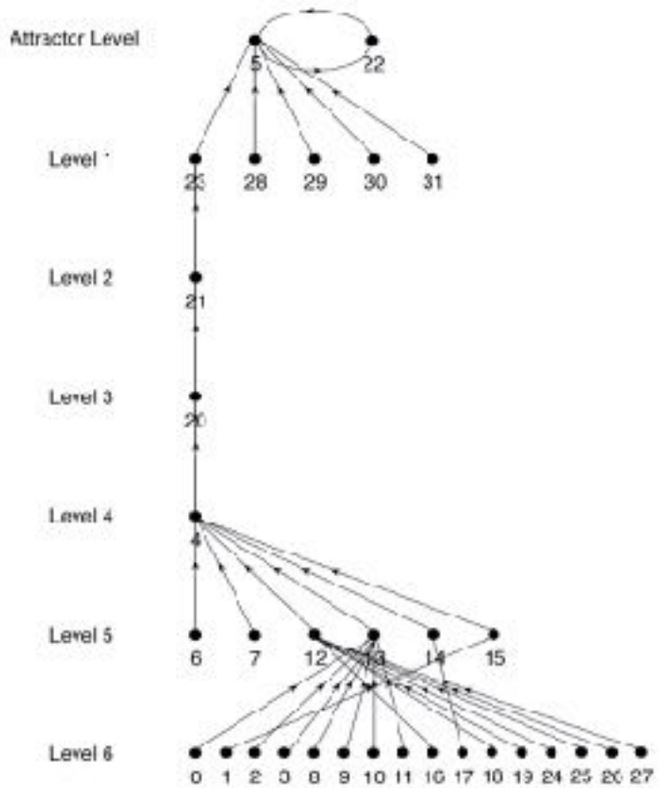
Neural network



Best-Fit



REVEAL



Summary

- We have shown that threshold BNs can be inferred from data using a neural network learning approach (perceptrons).

- An advantage in using the perceptron learning rule, in problems of threshold nature of course, is the existence of the perceptron convergence theorem (Bishop, 1995) which states that the learning rule will find a solution in a finite number of steps.

- Comparisons with the networks inferred using REVEAL and Best- Fit, showed that our method resembled in a better way the natural behaviour of bacteria population in terms of QS systems activation.

Thank you

This work was supported by Conicyt-Chile under grant Fondecyt 1180706