Reduction of Dynamical Systems over Monoids

G. Argyris, A. Lluch Lafuente, A. Leguizamon-Robayo, M. Tribastone,
Max Tschaikowski, and **Andrea Vandin**

DTU

AAU

IMT SCUOLA ALTI STUDI LUCCA

Sant'Anna School of Advanced Studies – Pisa

18/02/2025

Outline

- A bit of history
- Preliminaries
  - Discrete time systems and Boolean Networks
  - State Space + Dynamics = State Transition Graph
  - Attractors
  - State space explosion
  - T-Cell modelling
- Generalized Forward Bisimulation
- Biological Applications

RESEARCH ARTICLE | COMPUTER SCIENCES

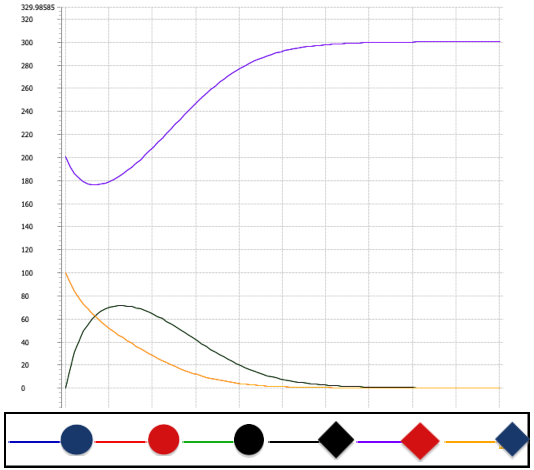
Maximal aggregation of polynomial dynamical systems

PNAS
Vol. 114 | No. 38

Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin Authors Info & Affiliations

Edited by Moshe Y. Vardi, Rice University, Houston, TX, and approved July 28, 2017 (received for review February 16, 2017)

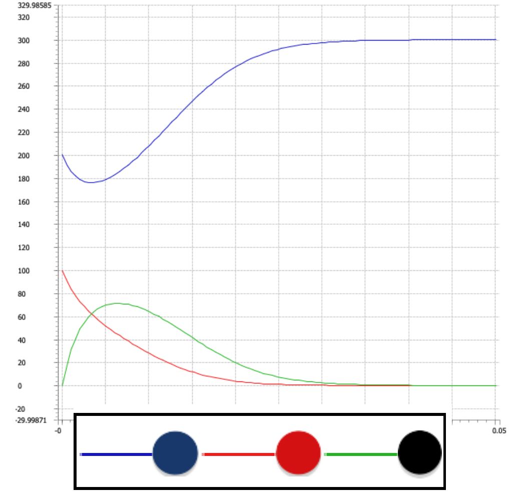
September 6, 2017 | 114 (38) 10029-10034 | https://doi.org/10.1073/pnas.1702697114



2 linear reductions for non-linear
Ordinary Differential Equations (ODEs)
Backward Reduction

$$\frac{dx(t)}{dt} = F(x(t)) \qquad x = (x_1, \ldots, x_n)$$

Backward reduction



RESEARCH ARTICLE | COMPUTER SCIENCES

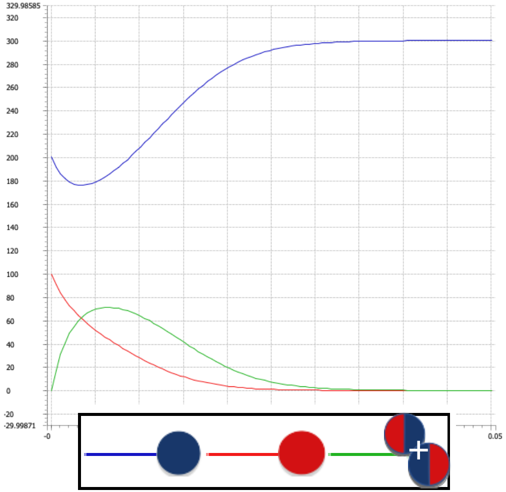
Maximal aggregation of polynomial dynamical systems

PNAS
Vol. 114 | No. 38

Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin Authors Info & Affiliations

Edited by Moshe Y. Vardi, Rice University, Houston, TX, and approved July 28, 2017 (received for review February 16, 2017)

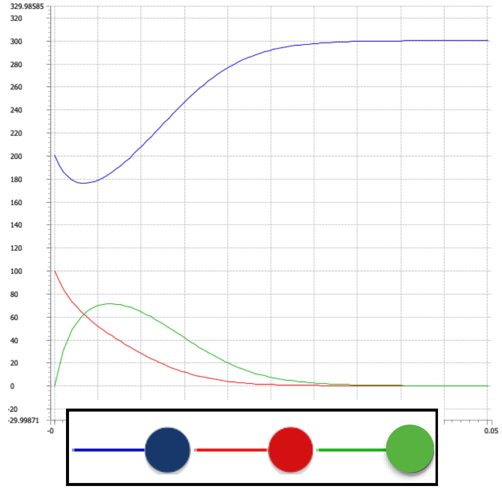
September 6, 2017 | 114 (38) 10029-10034 | https://doi.org/10.1073/pnas.1702697114



Forward reduction

2 linear reductions for non-linear
Ordinary Differential Equations (ODEs)
Forward Reduction (sum-preserving)

$$\frac{dx(t)}{dt} = F(x(t)) \qquad x = (x_1, \ldots, x_n)$$



RESEARCH ARTICLE | COMPUTER SCIENCES |

Maximal aggregation of polynomial dynamical systems

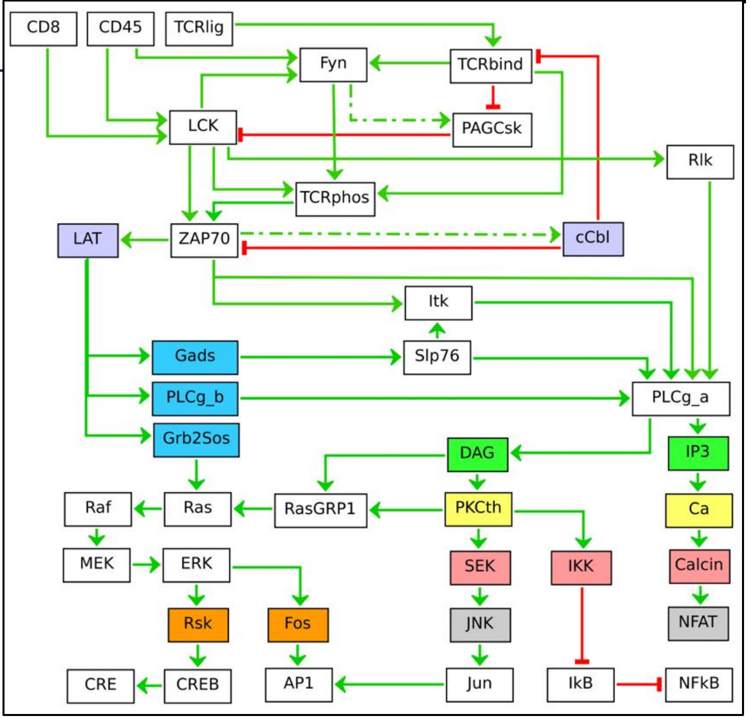
PNAS
Vol. 114 | No. 38

Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin Authors Info & Affiliations

Edited by Moshe Y. Vardi, Rice University, Houston, TX, and approved July 28, 2017 (received for review February 16, 2017)

September 6, 2017 | 114 (38) 10029-10034 | https://doi.org/10.1073/pnas.1702697114

Encodings allow to consider Boolean Networks (BNs)
With several limitations



2 linear reductions for non-linear
Ordinary Differential Equations (ODEs)
Forward Reduction (sum-preserving)

$$\frac{dx(t)}{dt} = F(x(t)) \qquad x = (x_1, \ldots, x_n)$$

Real expressions

Real variables

$$x(t+1) = F(x(t)) \qquad x = (x_1, \ldots, x_n)$$

Boolean expressions

Boolean variables

RESEARCH ARTICLE | COMPUTER SCIENCES

Maximal aggregation of polynomial dynamical systems

PNAS
Vol. 114 | No. 38

Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin Authors Info & Affiliations

Edited by Moshe Y. Vardi, Rice University, Houston, TX, and approved July 28, 2017 (received for review February 16, 2017)

September 6, 2017 | 114 (38) 10029-10034 | https://doi.org/10.1073/pnas.1702697114

Forward bisimulation ODEs:
Linear reduction (sum-preserving) of non-linear Ordinary Differential Equations (ODEs)

Minimization of Dynamical Systems over Monoids

Georgios Argyris*, Alberto Lluch Lafuente*, Alexander Leguizamon Robayo†, Mirco Tribastone‡, Max Tschaikowski†, and Andrea Vandin§,*

Thirty-Eighth Annual ACM/IEEE Symposium on
Logic in Computer Science (LICS)
Boston • 26–29 June 2023

Generalized Forward bisimulation:
Non-linear reduction (preserving **monoids operations**) of non-linear ODEs, **BNs**, **Difference Equations**,

More general family of models

More general reductions

RESEARCH ARTICLE | COMPUTER SCIENCES

Maximal aggregation of polynomial dynamical systems

PNAS
Vol. 114 | No. 38

Luca Cardelli, Mirco Tribastone, Max Tschaikowski, and Andrea Vandin Authors Info & Affiliations

Edited by Moshe Y. Vardi, Rice University, Houston, TX, and approved July 28, 2017 (received for review February 16, 2017)

September 6, 2017 | 114 (38) 10029-10034 | https://doi.org/10.1073/pnas.1702697114

Backward bisimulation ODEs:
Find variables that keep same solution if initialised equally for non-linear Ordinary Differential Equations (ODEs)

Research | Open Access | Published: 23 May 2023

Reducing Boolean networks with backward equivalence

Georgios A. Argyris, Alberto Lluch Lafuente, Mirco Tribastone, Max Tschaikowski & Andrea Vandin

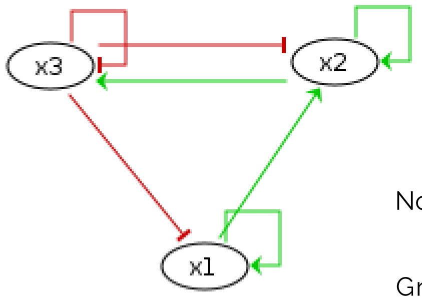
BMC Bioinformatics **24**, Article number: 212 (2023) | Cite this article

Boolean Backward equivalence:
Backward Reduction of BNs, partially asynchronous BNs

Different family of models

Boolean Network

Graphical representation

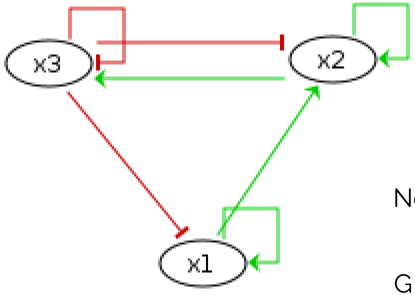


Nodes: Proteins, genes, molecules,
chemical compounds etc.

Green arrows: positive effect
Red arrows: negative effect

Boolean Network

Graphical representation



Nodes: Proteins, genes, molecules,
chemical compounds etc.

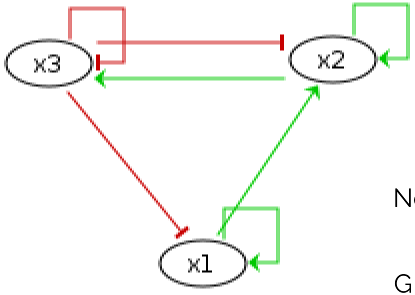
Green arrows: positive effect
Red arrows: negative effect

Boolean Network

$$
\begin{aligned}
x_1(t+1) &= \neg x_3(t) \vee x_1(t) \\
x_2(t+1) &= x_1(t) \vee x_2(t) \vee \neg x_3(t) \\
x_3(t+1) &= x_2(t) \wedge \neg x_3(t)
\end{aligned}
$$

Boolean Network

STG: State Transition Graph

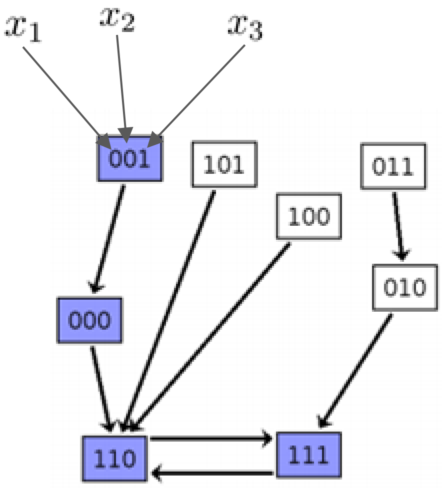


Nodes: Proteins, genes, molecules,
chemical compounds etc.

Green arrows: positive effect
Red arrows: negative effect

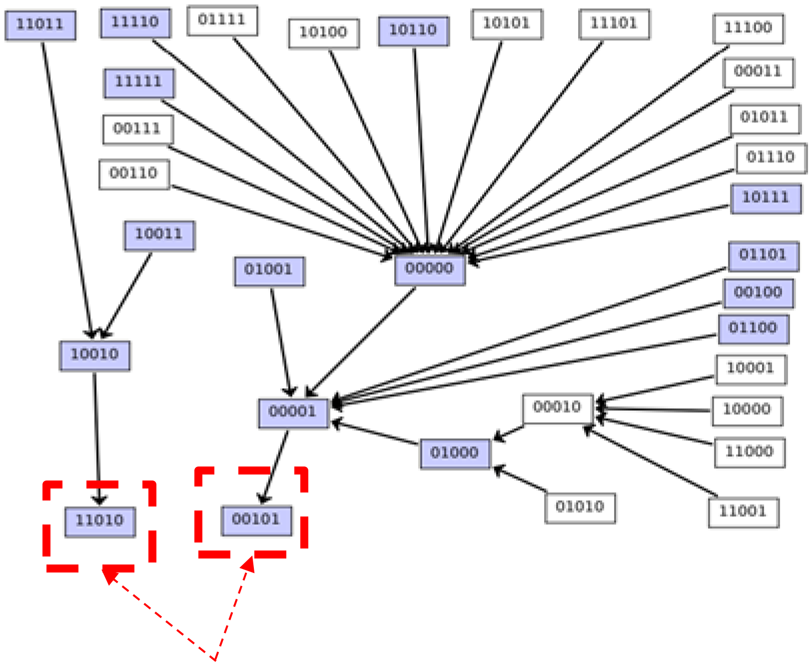
Boolean Network

$$
\begin{aligned}
x_1(t+1) &= \neg x_3(t) \vee x_1(t) \\
x_2(t+1) &= x_1(t) \vee x_2(t) \vee \neg x_3(t) \\
x_3(t+1) &= x_2(t) \wedge \neg x_3(t)
\end{aligned}
$$

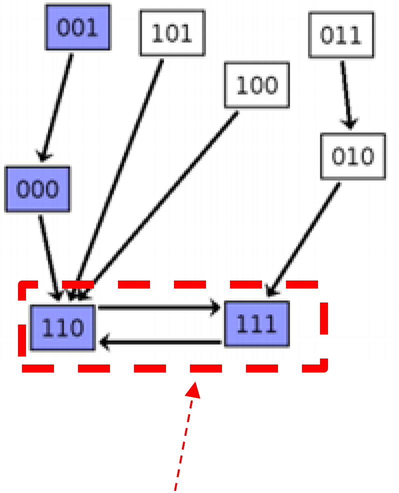


0: inactive
1: active

Attractors



Steady state attractor



Cyclic attractor

State Space Explosion

N variables ⇒ 2^N states

128 variables 2^{128} states

State Space Explosion

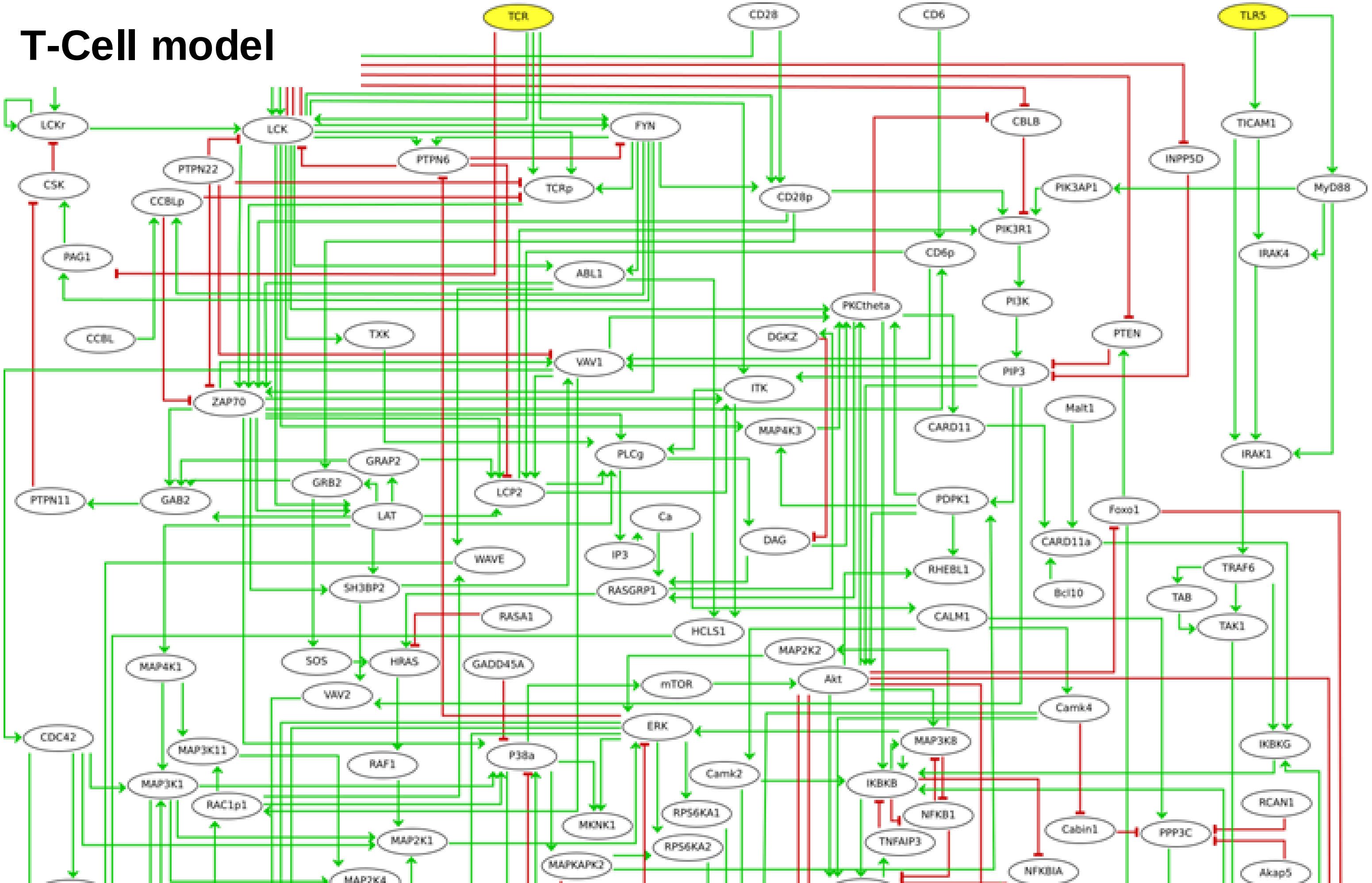
$$N \text{ variables} \Longrightarrow 2^N \text{ states}$$

$$128 \text{ variables} \qquad 2^{128} \text{ states}$$

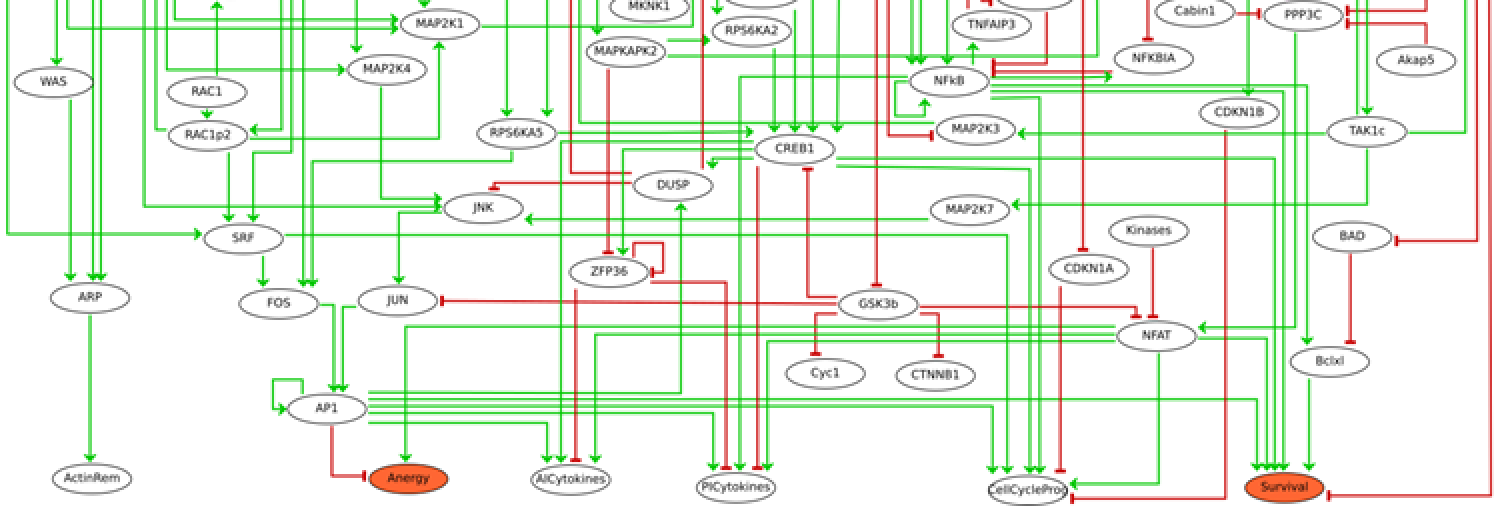
Reduction facilitates:

- STG generation
- Attractor computation
- Control
- ...

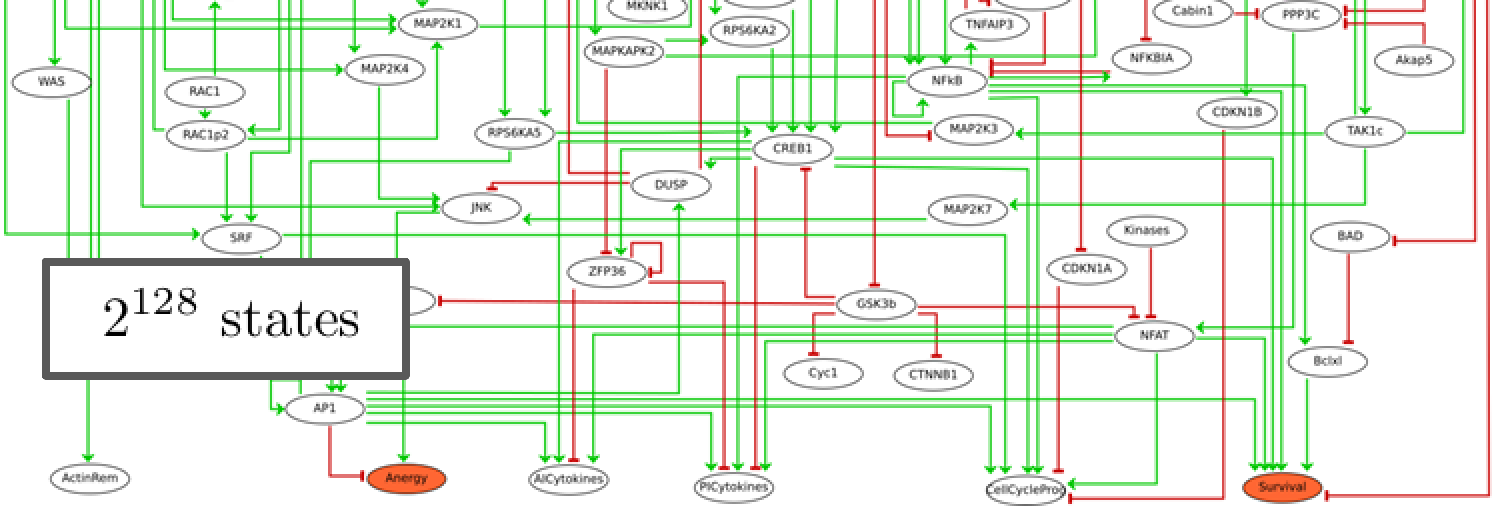
T-Cell model



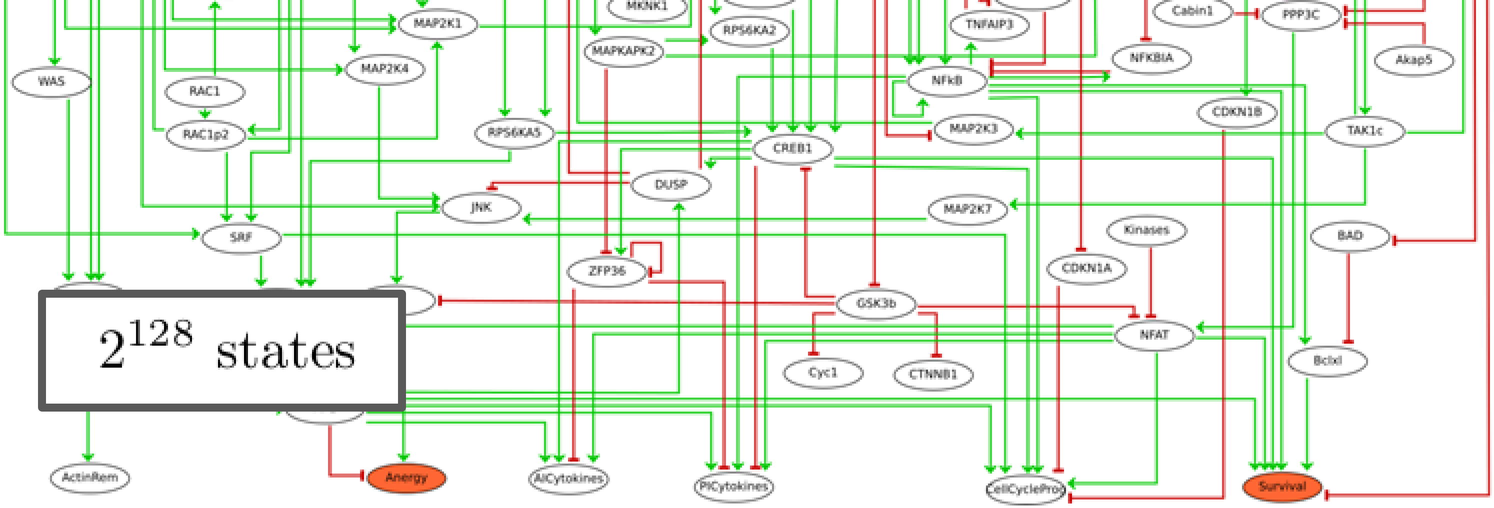
TCR
CD28
CD6
TLR5
LCKr
LCK
FYN
CBLB
TICAM1
PTPN6
INPP5D
PTPN22
CSK
TCRp
CD28p
PIK3AP1
MyD88
CC8Lp
PIK3R1
CD6p
IRAK4
PAG1
ABL1
PKCtheta
PI3K
CCBL
TXK
DGKZ
PTEN
VAV1
PIP3
ITK
ZAP70
Malt1
MAP4K3
CARD11
IRAK1
GRAP2
PLCg
GRB2
LCP2
PDPK1
PTPN11
GAB2
LAT
Ca
Foxo1
DAG
CARD11a
IP3
WAVE
RHEBL1
TRAF6
SH3BP2
RASGRP1
Bcl10
TAB
RASA1
CALM1
TAK1
HCLS1
MAP2K2
MAP4K1
SOS
HRAS
GADD45A
Akt
mTOR
VAV2
Camk4
ERK
CDC42
MAP3K8
IKBKG
MAP3K11
RAF1
P38a
Camk2
MAP3K1
IKBKB
RAC1p1
RPS6KA1
NFKB1
RCAN1
MKNK1
Cabin1
PPP3C
MAP2K1
RPS6KA2
TNFAIP3
MAPKAPK2
NFKBIA
Akap5
MAP2K4



MKNK1
MAP2K1
RPS6KA2
TNFAIP3
Cabin1
PPP3C
MAPKAPK2
NFKBIA
Akap5
WAS
MAP2K4
NFkB
RAC1
CDKN1B
RAC1p2
RPS6KA5
MAP2K3
TAK1c
CREB1
DUSP
JNK
MAP2K7
SRF
Kinases
BAD
ZFP36
CDKN1A
ARP
FOS
JUN
GSK3b
NFAT
Bclxl
Cyc1
CTNNB1
AP1
ActinRem
Anergy
AICytokines
PICytokines
CellCycleProl
Survival



MKNK1
MAP2K1
RPS6KA2
TNFAIP3
Cabin1
PPP3C
WAS
MAPKAPK2
MAP2K4
NFKBIA
Akap5
RAC1
NFkB
CDKN1B
RAC1p2
RPS6KA5
CREB1
MAP2K3
TAK1c
DUSP
JNK
MAP2K7
SRF
Kinases
BAD
ZFP36
CDKN1A
GSK3b
NFAT
Cyc1
CTNNB1
Bclxl
AP1
ActinRem
Anergy
AICytokines
PICytokines
Survival
2^{128} states



Model	*Size*	*Attractors*	*Analysis (s)*
Original T-Cell	128	—*Time Out*—	
Reduced T-Cell v1	116	—*Time Out*—	
Reduced T-Cell v2	98	8	9349.577
Reduced T-Cell v3	97	8	1103.912
Reduced T-Cell v4	95	2	29.336

Experiments with even larger BN: 321 variables

Model	*Variables*	*Attractors analysis*	
		Count	*Runtime(s)*
Original	321	—*Time Out*—	
Output separated	189	—*Time Out*—	
O1	70	64	0.668
O2	33	64	0.325
Maximal	1	1	0.001

- We consider large model (321 variables) by S.Raza,et al:

  A logic-based diagram of signalling pathways central to macrophage activation, **BMC systems biology**, 2008
- Original model could not be analysed
- Output-separated reduction is still too large, maximal reduction is trivial
- In O1 and O2 we admit to aggregate some outputs

  We can perform some analysis of an otherwise not analyzable model

  Reductions are useful!

Results on STG generation

Model	*Original model*		*Input-distinguished Reduced model*			*Maximal Reduced model*		
	Size	*STG generation(s)*	*Reduction (s)*	*Size*	*STG generation(s)*	*Reduction (s)*	*Size*	*STG generation(s)*
B7	33	*out of memory*	0.585	27	*out of memory*	0.608	25	*out of memory*
B9	28	*out of memory*	0.449	25	*out of memory*	0,416	20	52.8
B10	26	*out of memory*	0.227	23	457	0.145	4	0.006
B11	24	984	0.243	23	475	0.207	9	0,280
B12	24	987	0.349	21	102	0.121	4	0.050

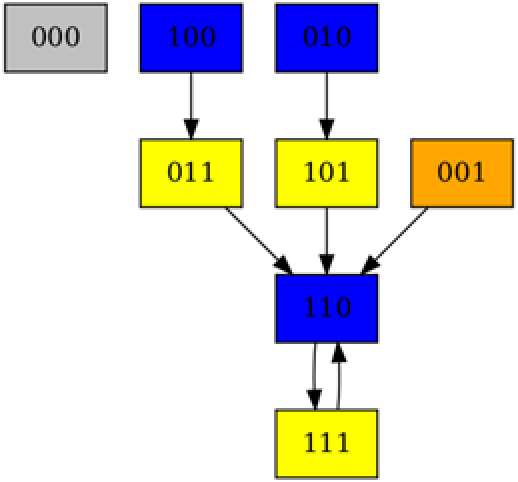
Example

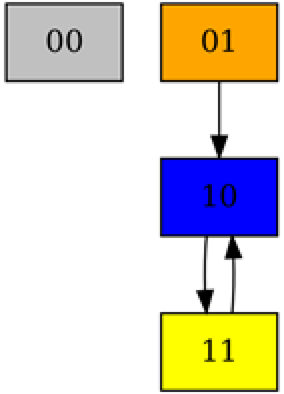
$$x_1(t+1) = x_2(t) \vee x_3(t)$$
$$x_2(t+1) = x_1(t) \vee x_3(t)$$
$$x_3(t+1) = \neg x_3(t) \wedge (x_1(t) \vee x_2(t))$$

Generalised Forward Bisimulation

$$x_{1,2} = x_1 \vee x_2$$

$$x_{1,2}(t+1) = x_{1,2}(t) \vee x_3(t)$$
$$x_3(t+1) = \neg x_3(t) \wedge x_{1,2}(t)$$





Generalized Forward Bisimulation

Merging of 2 variables

↓

Raise the definition of GFB to partition of variables

↓

Generalization of GFB over Boolean monoids

↓

Generalize GFB on Dynamical Systems over arbitrary Monoids

Boolean Network

Definition 1. *A Boolean network is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$.

$$
\begin{aligned}
f_{x_1} &= x_2 \vee x_3 \\
f_{x_2} &= x_1 \vee x_3 \\
f_{x_3} &= \neg x_3 \wedge (x_1 \vee x_2)
\end{aligned}
$$

Boolean Network

Definition 1. *A Boolean network is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$.

$$f_{x_1} = x_2 \vee x_3$$
$$f_{x_2} = x_1 \vee x_3$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2)$$

$$\downarrow$$

$$f_{x_1} \vee f_{x_2} = x_2 \vee x_3 \vee x_1 \vee x_3 = x_1 \vee x_2 \vee x_3$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2)$$

Boolean Network

Definition 1. *A Boolean network is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$.

$$f_{x_1} = x_2 \vee x_3$$
$$f_{x_2} = x_1 \vee x_3$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2)$$

$$\downarrow$$

$$f_{x_1} \vee f_{x_2} = x_2 \vee x_3 \vee x_1 \vee x_3 = x_1 \vee x_2 \vee x_3$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2)$$

$$\downarrow$$

$$f_{x_{1,2}} = x_3 \vee x_{1,2}$$
$$f_{x_3} = \neg x_3 \wedge x_{1,2}$$

Boolean Network

Definition 1. *A Boolean network is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$.

$$f_{x_1} = x_2 \vee x_3$$
$$f_{x_2} = x_1 \vee x_3$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2)$$

$$\downarrow$$

$$f_{x_1} \vee f_{x_2} = x_2 \vee x_3 \vee x_1 \vee x_3 = x_1 \vee x_2 \vee x_3 = f_{x_1} \vee f_{x_2}[x_1/0, x_2/x_1 \vee x_2]$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2) \qquad = f_{x_3}[x_1/0, x_2/x_1 \vee x_2]$$

$$\downarrow$$

$$f_{x_{1,2}} = x_3 \vee x_{1,2}$$
$$f_{x_3} = \neg x_3 \wedge x_{1,2}$$

Boolean Network

Definition 1. *A Boolean network is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$.

$$
\begin{aligned}
f_{x_1} &= x_2 \lor x_3 \\
f_{x_2} &= x_1 \lor x_3 \\
f_{x_3} &= \neg x_3 \land (x_1 \lor x_2)
\end{aligned}
$$

$\downarrow$

$$
\begin{aligned}
f_{x_1} \lor f_{x_2} &= x_2 \lor x_3 \lor x_1 \lor x_3 = x_1 \lor x_2 \lor x_3 = f_{x_1} \lor f_{x_2}[x_1/0, x_2/x_1 \lor x_2] \\
f_{x_3} &= \neg x_3 \land (x_1 \lor x_2) \qquad = f_{x_3}[x_1/0, x_2/x_1 \lor x_2]
\end{aligned}
$$

$\downarrow$

$$
\begin{aligned}
f_{x_{1,2}} &= x_3 \lor x_{1,2} \\
f_{x_3} &= \neg x_3 \land x_{1,2}
\end{aligned}
$$

Identity element of the monoid $(\mathbb{B}, \lor)$

Boolean Network

Definition 1. *A Boolean network is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$.

$$f_{x_1} = x_2 \vee x_3$$
$$f_{x_2} = x_1 \vee x_3$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2)$$

$$\downarrow$$

$$\boxed{f_{x_1} \vee f_{x_2}} = x_2 \vee x_3 \vee x_1 \vee x_3 = x_1 \vee x_2 \vee x_3 \boxed{= f_{x_1} \vee f_{x_2}[x_1/0, x_2/x_1 \vee x_2]}$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2) \qquad = f_{x_3}[x_1/0, x_2/x_1 \vee x_2]$$

$$\downarrow$$

$$f_{x_{1,2}} = x_3 \vee x_{1,2}$$
$$f_{x_3} = \neg x_3 \wedge x_{1,2}$$

Identity element of the monoid $(\mathbb{B}, \vee)$

Boolean Network

Definition 1. *A Boolean network is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$.

$$f_{x_1} = x_2 \vee x_3$$
$$f_{x_2} = x_1 \vee x_3$$
$$f_{x_3} = \neg x_3 \wedge (x_1 \vee x_2)$$

$$\boxed{f_{x_1} \vee f_{x_2}} = x_2 \vee x_3 \vee x_1 \vee x_3 = x_1 \vee x_2 \vee x_3 \boxed{= f_{x_1} \vee f_{x_2}[x_1/0, x_2/x_1 \vee x_2]}$$
$$\boxed{f_{x_3}} = \neg x_3 \wedge (x_1 \vee x_2) \qquad \boxed{= f_{x_3}[x_1/0, x_2/x_1 \vee x_2]}$$

Identity element of the monoid $(\mathbb{B}, \vee)$

$$f_{x_{1,2}} = x_3 \vee x_{1,2}$$
$$f_{x_3} = \neg x_3 \wedge x_{1,2}$$

Boolean Network

Definition 1. *A Boolean network is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{B}^n \to \mathbb{B}$.

$$\begin{aligned} f_{x_1} &= x_2 \vee x_3 \\ f_{x_2} &= x_1 \vee x_3 \\ f_{x_3} &= \neg x_3 \wedge (x_1 \vee x_2) \end{aligned}$$

$\downarrow$

$$\begin{aligned} f_{x_1} \vee f_{x_2} &= x_2 \vee x_3 \vee x_1 \vee x_3 = x_1 \vee x_2 \vee x_3 \\ f_{x_3} &= \neg x_3 \wedge (x_1 \vee x_2) \end{aligned}$$

$\downarrow$

$$\begin{aligned} f_{x_{1,2}} &= x_3 \vee x_{1,2} \\ f_{x_3} &= \neg x_3 \wedge x_{1,2} \end{aligned}$$

$$\boxed{\begin{aligned} f_{x_1} \vee f_{x_2} = f_{x_1} \vee f_{x_2}[x_1/0, x_2/x_1 \vee x_2] \\ \bigwedge f_{x_3} = f_{x_3}[x_1/0, x_2/x_1 \vee x_2] \end{aligned}}$$

Theorem:

- This SMT formula is valid IFF we can aggregate x1 and x2

Generalized Forward Bisimulation

Merging of 2 variables

↓

Raise the definition of GFB to partition of variables

↓

Generalization of GFB over Boolean monoids

↓

Generalize GFB on Dynamical Systems over arbitrary Monoids

Generalized Forward Bisimulation

Definition 2. *A partition* $P = \{P_1, P_2, \ldots\}$ *of the set of variables is a* Generalised Forward Bisimulation *if and only if the BN can be written in one variable for each block* P_i *of the partition, representing the disjunction (OR) of the variables belonging to this block.*

Generalized Forward Bisimulation

Definition 2. *A partition $P = \{P_1, P_2, \ldots\}$ of the set of variables is a* Generalised Forward Bisimulation *if and only if the BN can be written in one variable for each block P_i of the partition, representing the disjunction (OR) of the variables belonging to this block.*

$$
\begin{aligned}
f_{x_1} &= x_2 \vee x_3 \\
f_{x_2} &= x_1 \vee x_3 \\
f_{x_3} &= \neg x_3 \wedge (x_1 \vee x_2)
\end{aligned}
$$

$$P = \{\{x_1, x_2\}, \{x_3\}\}$$

Generalized Forward Bisimulation

Definition 2. *A partition $P = \{P_1, P_2, \ldots\}$ of the set of variables is a* Generalised Forward Bisimulation *if and only if the BN can be written in one variable for each block P_i of the partition, representing the disjunction (OR) of the variables belonging to this block.*

$$\begin{aligned} f_{x_1} &= x_2 \vee x_3 \\ f_{x_2} &= x_1 \vee x_3 \\ f_{x_3} &= \neg x_3 \wedge (x_1 \vee x_2) \end{aligned}$$

$$P = \{\{x_1, x_2\}, \{x_3\}\}$$

$$f_{x_1} \vee f_{x_2} = f_{x_1} \vee f_{x_2}[x_1/1, x_2/x_1 \vee x_2] \bigwedge f_{x_3} = f_{x_3}[x_1/0, x_2/x_1 \vee x_2]$$

The corresponding SMT formula for the partition

Generalized Forward Bisimulation

Definition 2. *A partition $P = \{P_1, P_2, \ldots\}$ of the set of variables is a* Generalised Forward Bisimulation *if and only if the BN can be written in one variable for each block P_i of the partition, representing the disjunction (OR) of the variables belonging to this block.*

$$
\begin{aligned}
f_{x_1} &= x_2 \vee x_3 \\
f_{x_2} &= x_1 \vee x_3 \\
f_{x_3} &= \neg x_3 \wedge (x_1 \vee x_2)
\end{aligned}
$$

$$P = \{\{x_1, x_2\}, \{x_3\}\}$$

$$f_{x_1} \vee f_{x_2} = f_{x_1} \vee f_{x_2}[x_1/1, x_2/x_1 \vee x_2] \bigwedge f_{x_3} = f_{x_3}[x_1/0, x_2/x_1 \vee x_2]$$

$\forall P_i, \forall x_i, x_j \in P_i$ the following formula holds:

$$\bigwedge_{P_i \in P} \Big(\bigvee_{x_k \in P_i} f_{x_k} = \bigvee_{x_k \in P_i} f_{x_k}[x_i/0][x_j/(x_i \vee x_j)] \Big)$$

Generalized Forward Bisimulation

Merging of 2 variables

↓

Raise the definition of GFB to partition of variables

↓

Generalization of GFB over Boolean monoids

↓

Generalize GFB on Dynamical Systems over arbitrary Monoids

Generalization to Boolean Monoids

$\forall P_i, \forall x_i, x_j \in P_i$ the following formula holds:

$$\bigwedge_{P_i \in P} \Big(\bigvee_{\boldsymbol{x_k \in P_i}} f_{x_k} = \bigvee_{\boldsymbol{x_k \in P_i}} f_{x_k}[x_i/\mathbf{0}][x_j/(x_i \vee x_j)] \Big)$$

Generalization to Boolean Monoids

$\forall P_i, \forall x_i, x_j \in P_i$ the following formula holds:

$$\bigwedge_{P_i \in P} \Big(\bigvee_{\boldsymbol{x_k \in P_i}} f_{x_k} = \bigvee_{\boldsymbol{x_k \in P_i}} f_{x_k}[x_i/\mathbf{0}][x_j/(x_i \vee x_j)] \Big)$$

For an arbitary commutative monoid $(\mathbb{B}, \oplus)$, the partition P is a Generalised Forward Bisimulation if and only if $\forall P_i, \forall x_i, x_j \in P_i$ the following formula holds:

$$\bigwedge_{P_i \in P} \Big(\bigoplus_{\boldsymbol{x_k \in P_i}} f_{x_k} = \bigoplus_{\boldsymbol{x_k \in P_i}} f_{x_k}[x_i/\mathbf{0}_{\oplus}][x_j/(x_i \oplus x_j)] \Big)$$

Generalized Forward Bisimulation

Merging of 2 variables

↓

Raise the definition of GFB to partition of variables

↓

Generalization of GFB over Boolean monoids

↓

Generalize GFB on Dynamical Systems over arbitrary Monoids

Definition 2. *A discrete-time dynamical system (DS) is a pair* (X, F) *where* $X = \{x_1, \ldots, x_n\}$ *is a set of variables, and* $F = \{f_{x_1}, \ldots, f_{x_n}\}$ *is a set of update functions with* $f_{x_i} : \mathbb{M}^n \to \mathbb{M}$ *being the update function of variable* x_i.

For an arbitary commutative monoid $(\mathbb{M}, \oplus)$, the partition P is a Generalised Forward Bisimulation if and only if $\forall P_i, \forall x_i, x_j \in P_i$ the following formula holds:

$$\bigwedge_{P_i \in P} \Big(\bigoplus_{\boldsymbol{x_k \in P_i}} f_{x_k} = \bigoplus_{\boldsymbol{x_k \in P_i}} f_{x_k}[x_i/\mathbf{0}_{\oplus}][x_j/(x_i \oplus x_j)] \Big)$$

$$(\mathbb{M}, \oplus) = (\mathbb{R}, +)$$
$$(\mathbb{M}, \oplus) = (\mathbb{R}, \cdot)$$

+ studied before, while * is new.
Example: Lotka-Volterra model

$$\begin{aligned}
\partial_t v_{x_1} &= v_{x_1}(1 - v_{x_2} v_{x_3}) \\
\partial_t v_{x_2} &= v_{x_2}(1 - v_{x_1}) \\
\partial_t v_{x_3} &= v_{x_3}(1 - v_{x_1})
\end{aligned}$$

has difference equations

$$\begin{aligned}
f_{x_1}(s) &= s_{x_1} + \tau s_{x_1}(1 - s_{x_2} s_{x_3}) \\
f_{x_2}(s) &= s_{x_2} + \tau s_{x_2}(1 - s_{x_1}) \\
f_{x_3}(s) &= s_{x_3} + \tau s_{x_3}(1 - s_{x_1})
\end{aligned}$$

that yield for $(\mathbb{M}, \oplus) = (\mathbb{R}, \cdot)$ the nonlinear reduction:

$$\begin{aligned}
\partial_t v_{x_1} &= v_{x_1}(1 - v_{x_2} v_{x_3}) \\
\partial_t (v_{x_2} v_{x_3}) &= 2 v_{x_1} v_{x_2}(1 - v_{x_1})
\end{aligned}$$

$$(\mathbb{M}, \oplus) = (\mathbb{Z}_n, min)$$
$$(\mathbb{M}, \oplus) = (\mathbb{Z}_n, max)$$

Multivalued Networks

$$x_1(t+1) = \begin{cases} 2 & \text{if } (1 \leq x_1(t) < 2) \vee ((x_3(t) \geq 1) \wedge (x_1(t) \geq 1)) \\ 1 & \text{if } (x_1(t) < 1) \wedge (x_3(t) \geq 1) \\ 0 & \text{otherwise} \end{cases}$$

$$x_2(t+1) = \begin{cases} 1 & \text{if } x_1(t) \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

$$x_3(t+1) = \begin{cases} 1 & \text{if } x_2(t) \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

Computation of the largest Bisimulation

Algorithm 1: Compute the largest GFB that refines the initial partition $\mathcal{X}_R$ for a DS (X, F).

Result: Largest GFB $\mathcal{H}$ that refines $\mathcal{X}_R$

$\mathcal{H} \leftarrow \mathcal{X}_R$;

while *true* **do**

  $\mathcal{H}' \leftarrow \emptyset$;

  for $H \in \mathcal{H}$ **do**

    $R \leftarrow \{(x_i, x_j) \in H \times H : \text{if } x_i \neq x_j,$ then $\Psi^{\mathcal{H}}_{x_i,x_j}$ and $\Psi^{\mathcal{H}}_{x_j,x_i}\}$;

    $\mathcal{H}' \leftarrow \mathcal{H}' \cup (H/R)$;

  end

  if $\mathcal{H} = \mathcal{H}'$ **then**

    return $\mathcal{H}$;

  else

    $\mathcal{H} \leftarrow \mathcal{H}'$;

  end

end

$$\Psi^{\mathcal{X}_R}_{x_i,x_j} \equiv \bigwedge_{C \in \mathcal{X}_R} \left(\bigoplus_{x_k \in C} f_{x_k} = \bigoplus_{x_k \in C} f_{x_k}[x_i/0_{\oplus}][x_j/(x_i \oplus x_j)] \right)$$

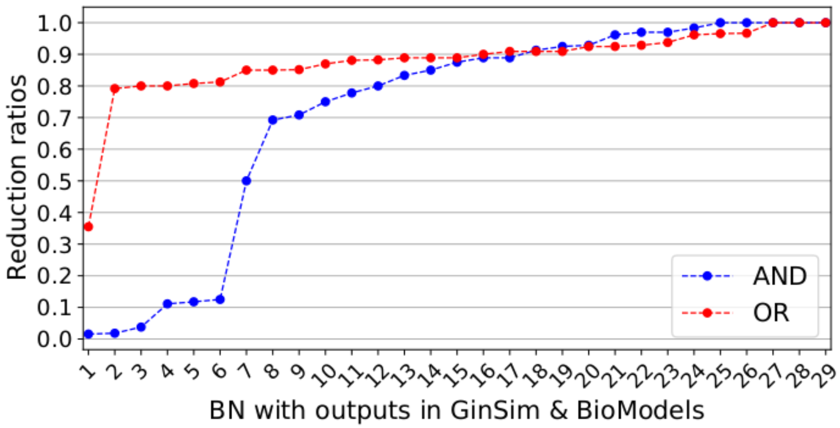
Experiments with GinSim & BioModels repositories of BNs

Do we get reductions in practice?

- We consider all 29 BN with outputs from 2 repo

  GinSim & BioModels
- Reduction ratio: reduced variables / original variables
- Moderate reduction power:

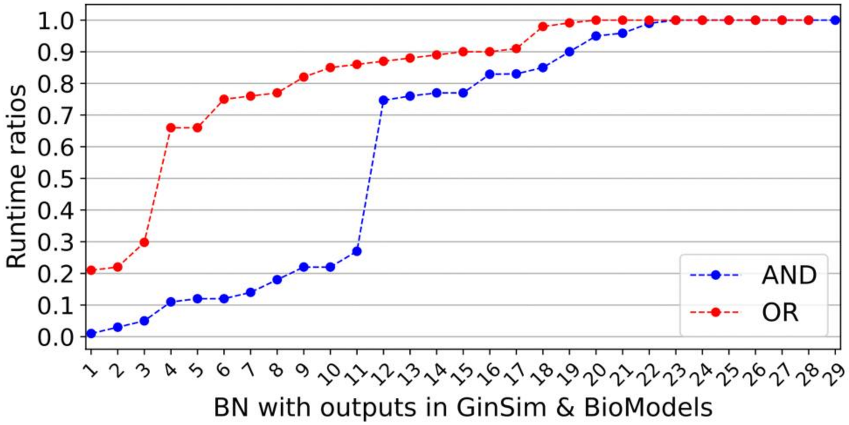
  For AND: 9 reductions have size less than 70% of original
- **Are these reductions useful?**



Yes, these reductions are useful

- We have run attractor analysis on the 29*3 models
- Runtime ratio: runtime reduced model / runtime reduced
- For AND, 11 models can be analysed faster

  In **less than 30% of the time of the original** ones



Experiments: Non-linear reduction of ODEs and Difference Eq.

Do we get **non-linear reductions in practice?**

- We consider 72 weighted networks up to 200 nodes

  From https://networks.skewed.de/

- We consider 2 dynamical interpretation for the adjacency matrix A

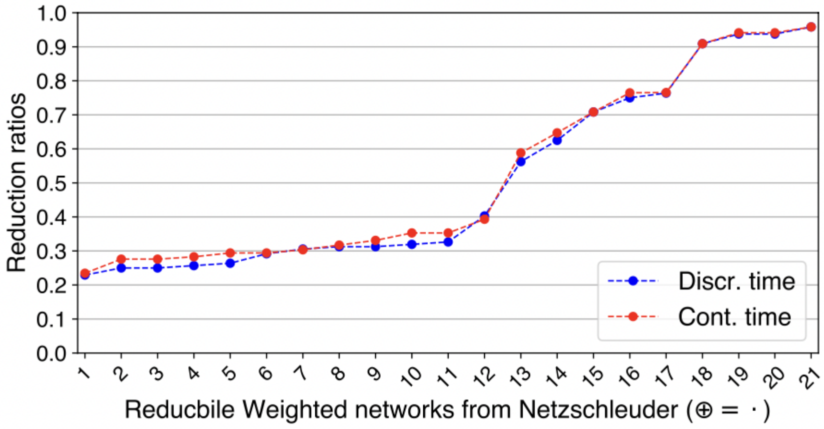
  Discrete-time: $x(t+1) = A(x(t))$

  Continuous-time: $\dfrac{dx(t)}{dt} = A(x(t))$

- Reduction ratio: original variables / reduced variables
- Good reduction power:

  We provide the **21 reducible models**

  12 models have less than 40% of the original number of variables



- Randomized polynomial time reduction algorithm for $(\mathbb{M}, \oplus) = (\mathbb{R}, \cdot)$
- Deterministic version suffers from exponential complexity and did not scale

Future Work

- On-the-fly computation of GFB
- Extension to hybrid dynamical systems
- Reduction of dynamical systems over arbitrary functions
- …

How do we reduce?

- GFB is implemented in the ERODE tool

www.erode.eu