Scalable and formal control of reachability in qualitative networks with PINT

Loïc Paulevé

CNRS/LRI, Univ. Paris-Sud, Univ. Paris-Saclay – BioInfo team

loic.pauleve@lri.fr

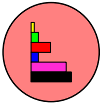
http://loicpauleve.name

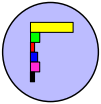
[BC]^2 workshop on logical modelling of biological regulatory networks

12 September 2017

Cellular Dynamics

Cell state
of interest

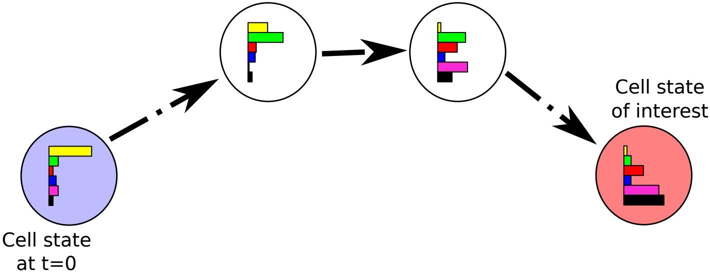




Cell state
at t=0

Initial state(s)/Goal state(s)

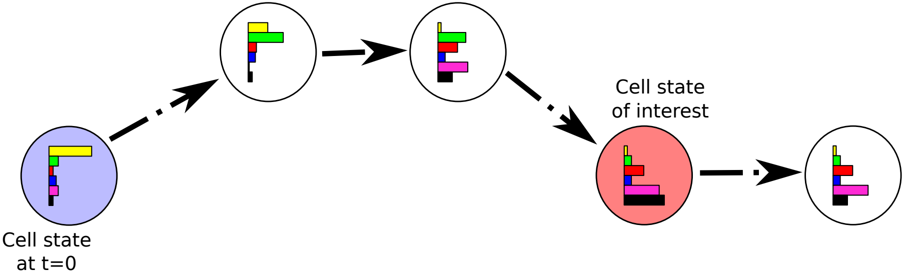
Cellular Dynamics



Initial state(s)/Goal state(s)

- Trajectory existence (reachability)

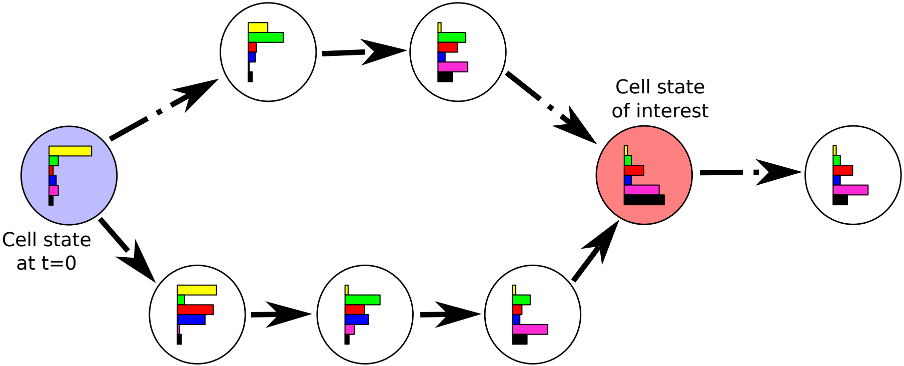
Cellular Dynamics



Initial state(s)/Goal state(s)

- Trajectory existence (reachability)

Cellular Dynamics



Initial state(s)/Goal state(s)

- Trajectory existence (reachability)
- Reasoning on all trajectories: e.g., **common features**
- Control: **perturbations** to avoid/enforce goal reachability

Analysis with the software PINT

Settings

- Qualitative models: state of components is Boolean/with a few levels
- Consider sets of initial conditions
- Goal states specified by the activity of narrow number of components (markers)

Analysis with the software PINT

Settings

- Qualitative models: state of components is Boolean/with a few levels
- Consider sets of initial conditions
- Goal states specified by the activity of narrow number of components (markers)

Contribution

- Avoid *in silico* screening of candidate perturbation
- Static analysis of traces: instead of computing (symbolic) state transition graph PINT computes a compact abstraction of it [Paulevé et al in Math Struct in Comp Sci 2012]
- Results are formal (guaranteed) but can be incomplete due to the abstraction

Analysis with the software PINT

Settings

- Qualitative models: state of components is Boolean/with a few levels
- Consider sets of initial conditions
- Goal states specified by the activity of narrow number of components (markers)

Contribution

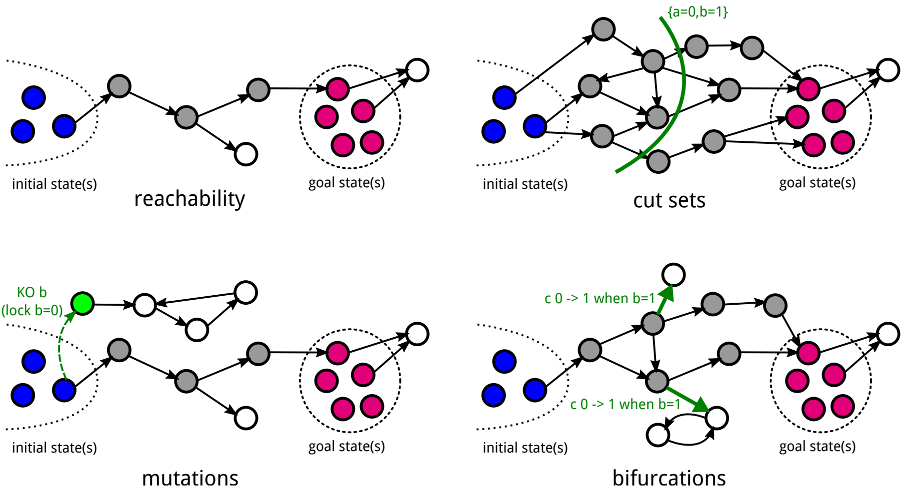
- Avoid *in silico* screening of candidate perturbation
- Static analysis of traces: instead of computing (symbolic) state transition graph PINT computes a compact abstraction of it [Paulevé et al in Math Struct in Comp Sci 2012]
- Results are formal (guaranteed) but can be incomplete due to the abstraction

Model input for PINT

- Main framework: automata networks (transition-centered specification)
- Boolean networks, multi-valued networks, 1-bounded Petri nets, . . .
- Uses BioLQM [Naldi et al] to import GINsim, SBML-Qual, BoolSim, . . .
  + other custom importations (BIOCHAM, SBGN-PD, . . .)

Remark: the following results do not depend on the updating schedule ((a)synchronous)

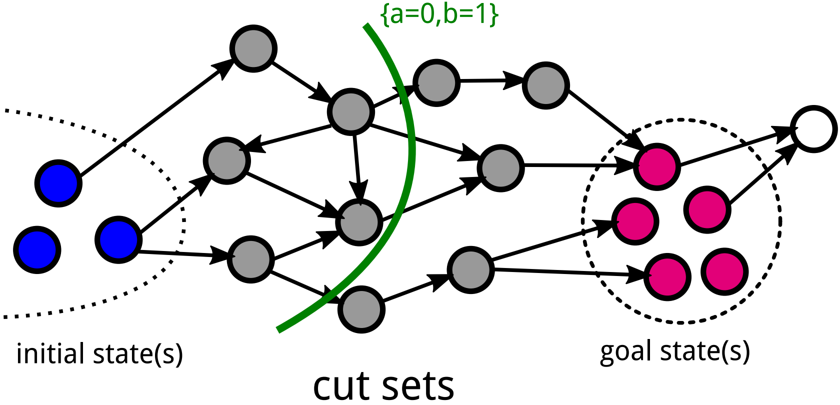
Static analysis of automata networks with Pint



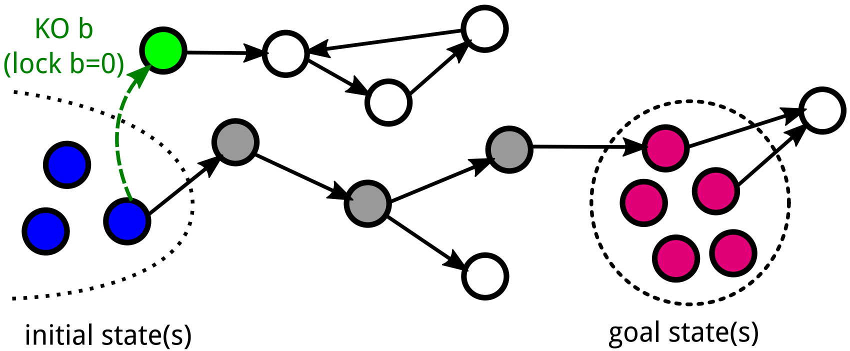
Software Pint - `http://loicpauleve.name/pint` [Paulevé at CMSB 2017]

Scalability: networks with 100 - 10,000 components

Static analysis of automata networks with Pint

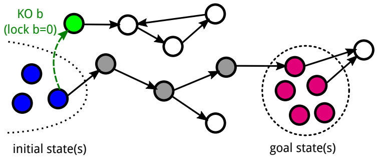


Static analysis of automata networks with Pint



mutations

Mutations



Definition

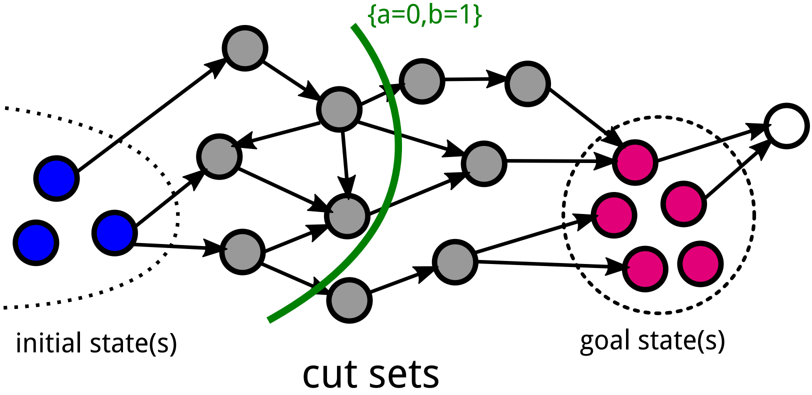
- Set of nodes with value to lock. E.g., $\{\mathbf{lock}(b = 0), \mathbf{lock}(c = 1)\}$.
- Goal is not reachable when applied to the initial state.

Implementation in PINT

- Answer-Set Programming (ASP) – similar to SAT.
- Encodes jointly the (abstracted) dynamics and the mutation property.
- Solutions are derived from the ASP logical model
  $\Rightarrow$ no screening.

(PINT approximates a PSPACE-complete problem into an NP problem)

Cut sets: common features of traces to goal

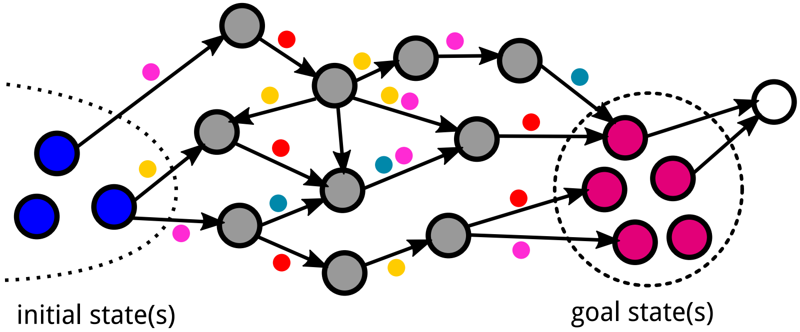


Interpretation: any trace requires at some point either $a = 0$ or $b = 1$.

Cut sets in the state transition graph

- From classical graph theory on labeled (multiple) directed graph
- Cut set = set of labels which **disconnects** initial states and goal states

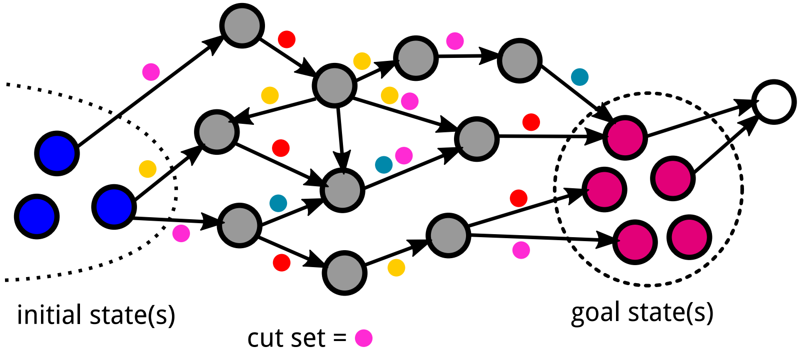
Labeled directed graph:



Cut sets in the state transition graph

- From classical graph theory on labeled (multiple) directed graph
- Cut set = set of labels which **disconnects** initial states and goal states

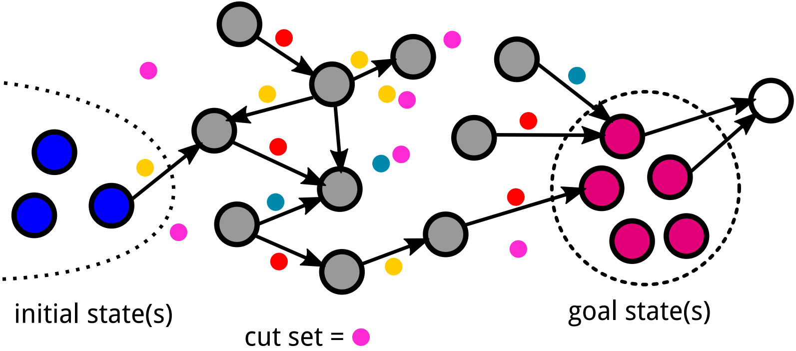
Labeled directed graph:



Cut sets in the state transition graph

- From classical graph theory on labeled (multiple) directed graph
- Cut set = set of labels which **disconnects** initial states and goal states

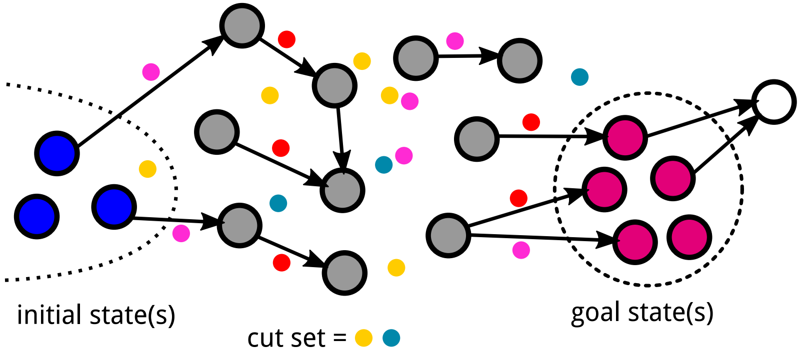
Labeled directed graph:



Cut sets in the state transition graph

- From classical graph theory on labeled (multiple) directed graph
- Cut set = set of labels which **disconnects** initial states and goal states

Labeled directed graph:

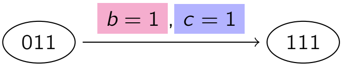


Transitions labels for Boolean networks

Labels = minimal cause of transitions

$$f_a(x) = x_b \text{ or } x_c$$

Labels for the transition $\langle 011\rangle \rightarrow \langle 111\rangle$



Remark: these labels come for free with automata networks/Petri nets

Transitions labels for Boolean networks

Labels = minimal cause of transitions

$$f_a(x) = x_b \text{ or } x_c$$

Labels for the transition $\langle 011 \rangle \rightarrow \langle 111 \rangle$

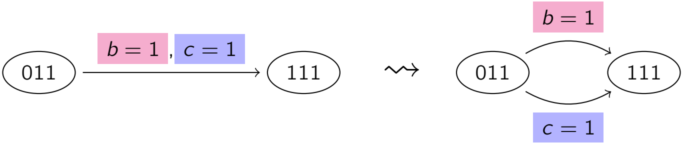
Remark: these labels come for free with automata networks/Petri nets

Transitions labels for Boolean networks

Labels = minimal cause of transitions

$$f_a(x) = x_b \text{ or } x_c$$

Labels for the transition $\langle 011\rangle \rightarrow \langle 111\rangle$



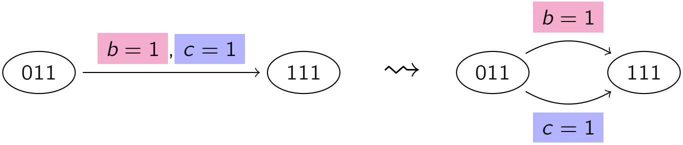
Remark: these labels come for free with automata networks/Petri nets

Transitions labels for Boolean networks

Labels = minimal cause of transitions

$$f_a(x) = x_b \text{ or } x_c$$

Labels for the transition $\langle 011 \rangle \rightarrow \langle 111 \rangle$



$\Rightarrow$ **1 cut set** to disconnect $\langle 011 \rangle$ and $\langle 111 \rangle$: $\{ b = 1, c = 1 \}$.

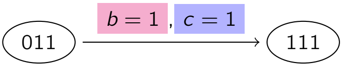
Remark: these labels come for free with automata networks/Petri nets

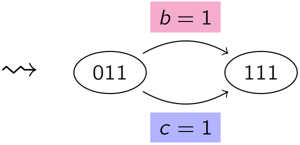
Transitions labels for Boolean networks

Labels = minimal cause of transitions

$$f_a(x) = x_b \text{ or } x_c$$

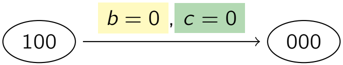
Labels for the transition $\langle 011\rangle \rightarrow \langle 111\rangle$





⇒ **1 cut set** to disconnect $\langle 011\rangle$ and $\langle 111\rangle$: $\{ b = 1, c = 1 \}$.

Labels for the transition $\langle 100\rangle \rightarrow \langle 000\rangle$



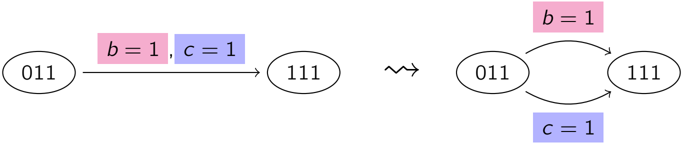
Remark: these labels come for free with automata networks/Petri nets

Transitions labels for Boolean networks

Labels = minimal cause of transitions

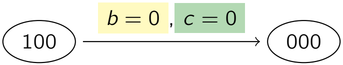
$$f_a(x) = x_b \text{ or } x_c$$

Labels for the transition $\langle 011 \rangle \rightarrow \langle 111 \rangle$



$\Rightarrow$ **1 cut set** to disconnect $\langle 011 \rangle$ and $\langle 111 \rangle$: $\{ b = 1, c = 1 \}$.

Labels for the transition $\langle 100 \rangle \rightarrow \langle 000 \rangle$



$\Rightarrow$ **2 cut sets** to disconnect $\langle 100 \rangle$ and $\langle 000 \rangle$: $\{ b = 0 \}$, $\{ c = 0 \}$.

Remark: these labels come for free with automata networks/Petri nets

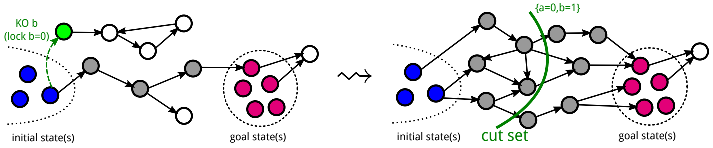
Cut sets in PINT

Implementation

- Does not build the state transition graph.
- Dedicated algorithm on its abstraction (3-partite graph) [Paulevé et al at CAV 2013].
- Solutions are derived, no candidate enumeration (no screening).

An alternative implementation with ASP could be considered as well.

Mutations vs Cut sets



From mutations to cut sets

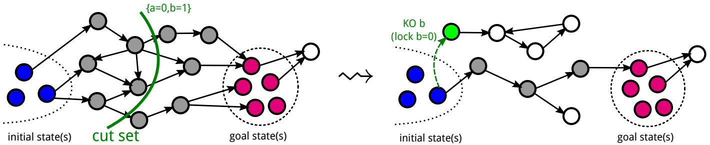
- lock($a = i$) cuts transitions using $a = j$, $j \neq i$
- To any mutation set corresponds a cut set
- ... *but* for a potentially different initial state

E.g., consider the mutation set

$$\begin{aligned} & \{\text{lock}(b = 0), \text{lock}(c = 1)\} \\ \Rightarrow \quad & \{b = 1, c = 0\} \end{aligned}$$

is a cut set for the goal from the initial state with $b = 0$ and $c = 1$.

Mutations vs Cut sets



From cut sets to mutations

- Idea: if every trace to goal uses either $a = 1$ or $c = 1$,
  KO of a and b prevents goal reachability
- ...*but* a cut set can include both active and inactive state of a same node
- ...the mutation should not change the initial state.

$\Rightarrow$ a cut set does not always gives a mutation

E.g., considering the initial state $\langle a = 0, b = 1, c = 0 \rangle$,

- $\{a = 1, c = 1\}$ gives guaranteed mutation $\{\text{lock}(a = 0), \text{lock}(c = 0)\}$;
- $\{a = 1, a = 0, b = 0\}$ gives no mutation;
- $\{a = 1, b = 1\}$ gives non-proven mutation $\{\text{lock}(a = 0), \text{lock}(b = 0)\}$.

Scalability of implementations

Computations with PINT:

- *k*-cut sets: OCaml implementation (algorithm on abstract structure)
- *k*-mutations: Answer-Set Programming (ASP) with clingo

k: limit results to sets with at most *k* elements.

Goal	TCell-d (101) FOXP3=1		MAPK (309) ERK-PP=1		PID (10,229) SNAIL=1	
3-cut sets	0.06s	35	0.06s	24	1.2s	7
4-cut sets	0.10s	101	0.1s	48	5s	37
6-cut sets	0.60s	495	1s	60	10m	907
3-mutations	0.30s	15	5s	222	50m	7
4-mutations	0.30s	15	10s	1896	50m	67
6-mutations	0.30s	15	Too many		50m	367

TCell-d: Abou-Jaoudé et al. in Frontiers in Bioengin. and Biotech, 2015
MAPK: Schoeberl et al. in Nature biotech, 2002
PID: Pathway interaction database full extraction (Paulevé et al. at CAV 2013).

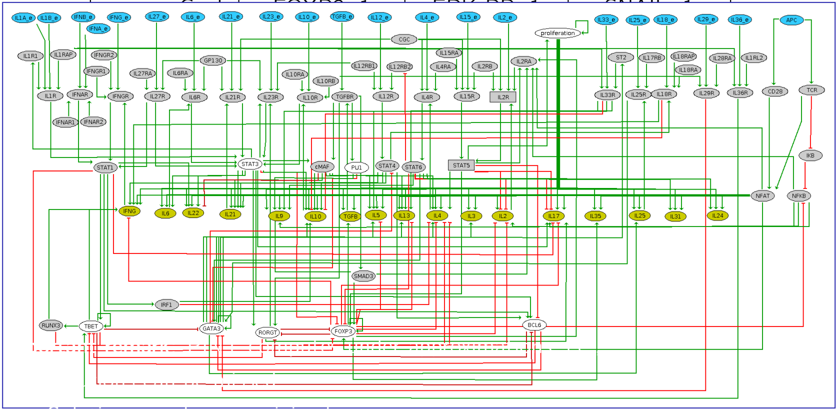
Scalability of implementations

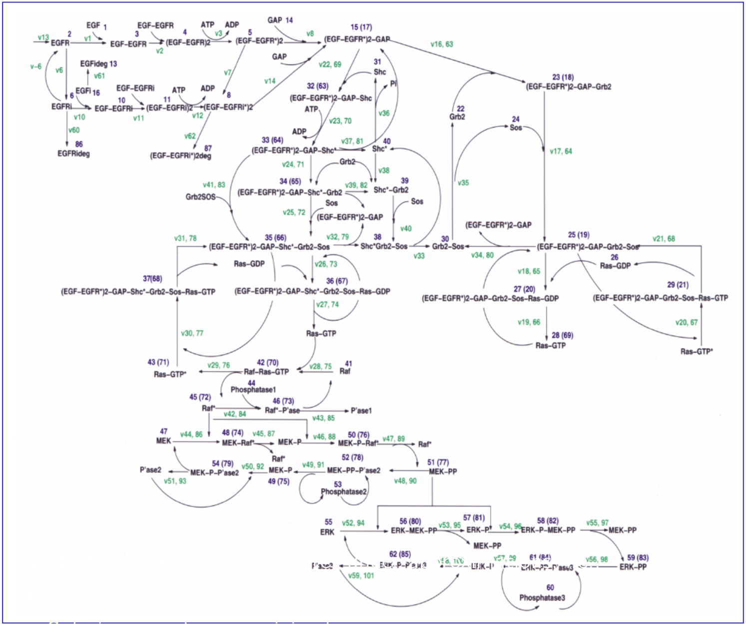
Computations with PINT:

- *k*-cut sets: OCaml implementation (algorithm on abstract structure)
- *k*-mutations: Answer-Set Programming (ASP) with clingo

k: limit results to sets with at most *k* elements.

	TCell-d (101)	MAPK (309)	PID (10,229)




EGF 1
EGF–EGFR
ATP
ADP
GAP 14
v13
2 EGFR
v1
3 EGF–EGFR
4 (EGF–EGFR)2
v2
v3
5 (EGF–EGFR*)2
v8
15 (17) (EGF–EGFR*)2–GAP
v16, 63
GAP
v–6
v6
EGFIdeg 13
v61
EGFI 16
6 EGFRi
EGF–EGFRi
10 EGF–EGFRi
11 (EGF–EGFRi)2
v10
v11
v12
8 (EGF–EGFRi*)2
v7
v14
v22, 69
31 Shc
Pi
32 (63) (EGF–EGFR*)2–GAP–Shc
ATP
ADP
v23, 70
v36
23 (18) (EGF–EGFR*)2–GAP–Grb2
22 Grb2
24 Sos
v17, 64
v60
86 EGFRideg
v62
87 (EGF–EGFRi*)2deg
33 (64) (EGF–EGFR*)2–GAP–Shc*
v37, 81
40 Shc*
v24, 71
Grb2
v38
v41, 83
Grb2SOS
34 (65) (EGF–EGFR*)2–GAP–Shc*–Grb2
v39, 82
39 Shc*–Grb2
Sos
v25, 72
(EGF–EGFR*)2–GAP
v40
v35
(EGF–EGFR*)2–GAP
v31, 78
35 (66) (EGF–EGFR*)2–GAP–Shc*–Grb2–Sos
v32, 79
38 Shc*Grb2–Sos
v33
30 Grb2–Sos
v34, 80
25 (19) (EGF–EGFR*)2–GAP–Grb2–Sos
v21, 68
Ras–GDP
v26, 73
26 Ras–GDP
v18, 65
37(68) (EGF–EGFR*)2–GAP–Shc*–Grb2–Sos–Ras–GTP
36 (67) (EGF–EGFR*)2–GAP–Shc*–Grb2–Sos–Ras–GDP
27 (20) (EGF–EGFR*)2–GAP–Grb2–Sos–Ras–GDP
29 (21) (EGF–EGFR*)2–GAP–Grb2–Sos–Ras–GTP
v27, 74
v19, 66
v20, 67
v30, 77
Ras–GTP
28 (69) Ras–GTP
Ras–GTP*
43 (71) Ras–GTP*
v29, 76
42 (70) Raf–Ras–GTP
v28, 75
41 Raf
44 Phosphatase1
45 (72) Raf*
46 (73) Raf*–P'ase
P'ase1
v42, 84
v43, 85
47 MEK
v44, 86
48 (74) MEK–Raf*
v45, 87
MEK–P
v46, 88
50 (76) MEK–P–Raf*
v47, 89
Raf*
Raf*
52 (78) MEK–PP–P'ase2
51 (77) MEK–PP
P'ase2
54 (79) MEK–P–P'ase2
v50, 92
MEK–P
v49, 91
49 (75)
v48, 90
v51, 93
53 Phosphatase2
55 ERK
v52, 94
56 (80) ERK–MEK–PP
v53, 95
57 (81) ERK–P
v54, 96
58 (82) ERK–P–MEK–PP
v55, 97
MEK–PP
MEK–PP
62 (85)
v57, 99
61 (84)
v56, 98
59 (83) ERK–PP
v59, 101
60 Phosphatase3

Scalability of implementations

Computations with PINT:

- *k*-cut sets: OCaml implementation (algorithm on abstract structure)
- *k*-mutations: Answer-Set Programming (ASP) with clingo

k: limit results to sets with at most *k* elements.

Goal	TCell-d (101) FOXP3=1		MAPK (309) ERK-PP=1		PID (10,229) SNAIL=1	
3-cut sets	0.06s	35	0.06s	24	1.2s	7
4-cut sets	0.10s	101	0.1s	48	5s	37
6-cut sets	0.60s	495	1s	60	10m	907
3-mutations	0.30s	15	5s	222	50m	7
4-mutations	0.30s	15	10s	1896	50m	67
6-mutations	0.30s	15	Too many		50m	367

TCell-d: Abou-Jaoudé et al. in Frontiers in Bioengin. and Biotech, 2015
MAPK: Schoeberl et al. in Nature biotech, 2002
PID: Pathway interaction database full extraction (Paulevé et al. at CAV 2013).

Important remark: PINT computes under-approximations, i.e.,

- Solutions can be missed
- Solutions can be non-minimal

Demo

There is a public instance of Jupyter/pint at
`http://tmpnb.loicpauleve.name`

Be aware that

- For short tests only (limited session durations)
- It runs in a virtual machine with limited resources on a personal server

You can install Pint locally either with binary packages or using docker
`http://loicpauleve.name/pint/doc/`

Conclusion

Mutations vs Cut sets from initial states to goal states

- Mutations: block reachability of the goal
- Cut sets: common features of all the trajectories to the goal
- Mutations give cut sets, but (some are) for different initial states
- Some cut sets cannot be transformed into mutations
  (if they change the initial state; if they use both the active and inactive form of a node)

PINT implementation

- No screening of candidates, solutions are deduced by static analysis
- Scalable to networks >1,000 components
- Under-approximation (no magic)

On-going and future work

- Compute temporal mutations
- Static quantification of the effect of a partial mutation/cut sets

Acknowledgements

LS2N, Nantes MeForBio

- Olivier Roux
- Morgan Magnin
- Maxime Folschette

Institut Curie

- Laurence Calzone
- Andrei Zinovyev

IRISA, Rennes Dyliss

- Geoffroy Andrieux

ETH Zürich BISON (now TU Darmstadt)

- Heinz Koeppl

Funding

- ANR-FNR "AlgoReCell" project - `http://algorecell.lri.fr` (2017-2020)
- CNRS PEPS INS2I"FoRCe" project (2017)