boolean_dsgrn

July 14, 2025

[1]:
```
import ginsim
import biolqm
import boolsim
import pandas as pd
from colomoto_jupyter import tabulate
from colomoto.minibn import BooleanNetwork
from colomoto.temporal_logics import *
```

This notebook has been executed using the docker image `colomoto/colomoto-docker:2025-03-01`

0.1 boolsim

[2]:
```
## Define network
# network = {
#     "ATM": "!Wip1",
#     "Chk2": "ATM & !Wip1",
#     "Mdm2": "p53",
#     "p53": "(ATM | Chk2) & !Mdm2",
#     "Wip1": "p53"
# }
network = {
    "x" : "!y",
    "y" : "!x"}
bn = BooleanNetwork(network)
```

[3]:
```
## Tabulate the attractors using boolsim
%time A = boolsim.attractors(bn)
tabulate(A)
```

```
CPU times: user 24.7 ms, sys: 5.11 ms, total: 29.8 ms
Wall time: 116 ms
```

[3]:

	x	y
0	0	1
1	1	0

[4]:
```
## Print attractors. The asterisk means "complex" attractors
for att in A:
    print(att)
```

```
{'x': 0, 'y': 1}
{'x': 1, 'y': 0}
```

1 bioLQM

[5]:
```
## Convert from boolsim to biolqm
lqm = bn.to_biolqm()
```

[6]:
```
## Save for future reference
biolqm.save(lqm, "network.net", "boolsim")
```

[6]:
```
'network.net'
```

[7]:
```
# Influence graph from biolqm
biolqm.influence_graph(lqm)
```

```
# computing graph layout…
```

[7]:
```
<networkx.classes.multidigraph.MultiDiGraph at 0x7fffcb1cf010>
```

1.0.1 Identification of stable states (fixed points)

[8]:
```
fps = biolqm.fixpoints(lqm)
pd.DataFrame(fps)
```

[8]:

	x	y
0	0	1
1	1	0

1.0.2 Identification of stable motifs (trapspaces)

A stable motif (also called symbolic steady state) is a partially assigned state such that all possible successors of all states which belong to the motif also belong to the motif.

[9]:
```
traps = biolqm.trapspace(lqm)
pd.DataFrame(traps)
```

[9]:

	x	y
0	1	0
1	0	1

1.0.3 Stable (states?)

[10]:
```
states = biolqm.stable(lqm)
print(states)
```

```
[{'x': 0, 'y': 1}, {'x': 1, 'y': 0}]
```

There are some funcionalities regarding "simulation" of deterministic paths (and non-deterministic by doing random walks).

1.0.4 Model perturbation

the biolqm.perturbation function enables the construction of a variant of the model, where the logical function of one (or several) component has been modified. A textual parameter describes the modification:

component%0 defines a knockout of a component component%1 defines an ectopic expression component%1:2 restricts the range of values for multi-valued components regulator:component%0 allows to remove a regulator In the following, we show the impact of the ectopic expression of the CycD component on the stable states and trapspaces on the model.

[11]:
```
pert = biolqm.perturbation(lqm, "y%1")
```

[12]:
```
fps = biolqm.fixpoints(pert)
pd.DataFrame(fps)
```

[12]:

	x	y
0	0	1

[13]:
```
traps = biolqm.trapspace(pert, "terminal")
pd.DataFrame(traps)
```

[13]:

	x	y
0	0	1

2 GINsim

[14]:
```
## Convert from biolqm to ginsim
lrg = biolqm.to_ginsim(lqm)
ginsim.show(lrg)
```

[14]: <IPython.core.display.HTML object>

[15]:
```
## Fixed points
fps = biolqm.fixpoints(lqm)
print(len(fps), "fixpoints")
```

```
2 fixpoints
```

[16]:
```
# First fixed point
ginsim.show(lrg, fps[0])
```

[16]: <IPython.core.display.HTML object>

[17]:
```
# Second fixed point
ginsim.show(lrg, fps[1])
```

[17]: <IPython.core.display.HTML object>

[ ]: