Formal modelling and analysis of ecosystems

Franck Pommereau[1] Cédric Gaucherel[2]

William Atger[1] Colin Thomas[1,2]

[1]IBISC, University of Évry / Paris-Saclay

[2]AMAP—INRA, CIRAD, Montpellier

Outline

Earth's 6th massive extinction event under way

EXAMPLES OF DECLINES IN NATURE

ECOSYSTEM EXTENT AND CONDITION

47%

Natural ecosystems have **declined by 47 per cent** on average, relative to their earliest estimated states.

SPECIES EXTINCTION RISK

25%

Approximately **25 per cent of species are already threatened with extinction** in most animal and plant groups studied.

ECOLOGICAL COMMUNITIES

23%

Biotic integrity—the abundance of naturally-present species—has **declined by 23 per cent** on average in terrestrial communities.*

BIOMASS AND SPECIES ABUNDANCE

82%

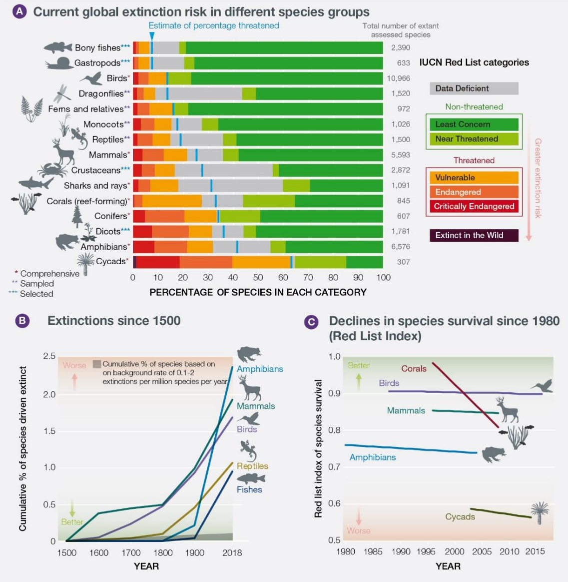
The global biomass of wild mammals has **fallen by 82 per cent.*** Indicators of vertebrate abundance have declined rapidly since 1970

NATURE FOR INDIGENOUS PEOPLES AND LOCAL COMMUNITIES

72%

72 per cent of indicators developed by indigenous peoples and local communities show **ongoing deterioration** of elements of nature important to them

* Since prehistory



Summary for policymakers of the global assessment report on biodiversity and ecosystem services — IPBES — May 2019

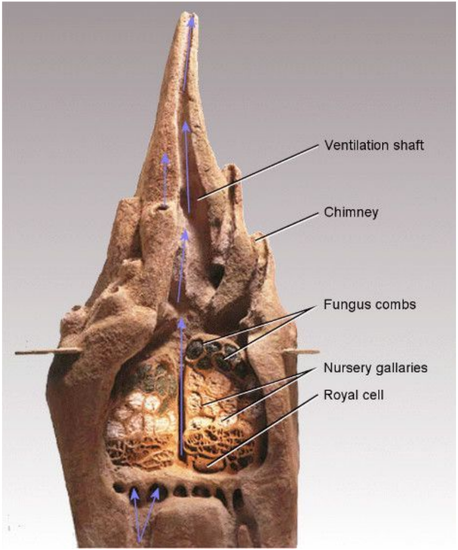
Understanding ecosystems ⇒ actions for their conservation

- formal modelling and analysis
- dynamics understanding
  - tipping points
  - catastrophic shifts
  - causality
  - paths to recovery
- abstraction ⇒ extract "laws"
  - discrete modelling
  - qualitative analysis
- confront with "in the field" studies (& experiments)

Christoph Niemann, NYT

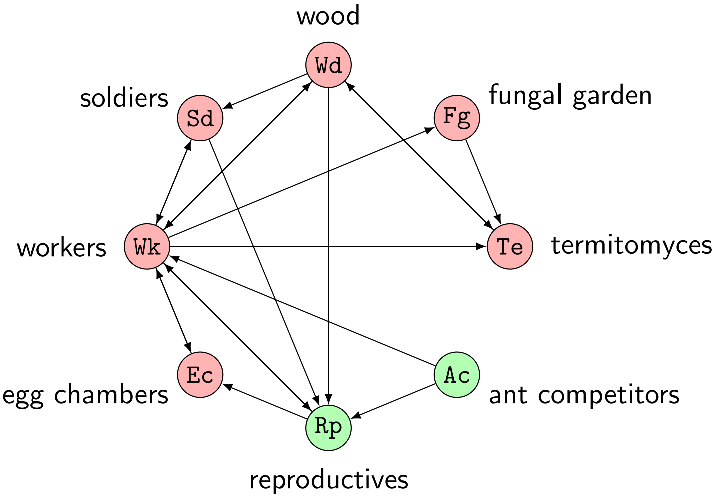
Outline

Running example: a termites colony



Ecosystemic graph

Aka interaction/influence graph/network



Entities, rules, constraints

Reaction Rules formalism (RR)

```
inhabitants:
  Rp+:  reproductives
  Wk-:  workers
  Sd-:  soldiers
  Te-:  termitomyces
structures:
  Ec-:  egg chambers
  Fg-:  fungal gardens
resources:
  Wd-:  wood
competitors:
  Ac+:  ant competitors
```

```
constraints:
  Fg- >> Te-
rules:
  Rp+ >> Ec+
  Rp+, Ec+ >> Wk+
  Wk+ >> Wd+, Te+, Fg+, Ec+
  Wk+, Wd+ >> Sd+, Rp+
  Wk+, Te+ >> Wd-
  Wd- >> Wk-, Te-
  Wk- >> Fg-, Sd-
  Wk-, Rp- >> Ec-
  Ac+, Sd- >> Wk-, Rp-
```

Constraints are rules with a higher priority:

- no fungal garden $\Rightarrow$ no fungi
- define transient states

Semantics

Proposition: Boolean networks $\subsetneq$ reaction rules = all Boolean LTS

Operational:

- state = entities valuation
- transition = application of a rule/constraint
- constraints have a higher priority
- no side-loops

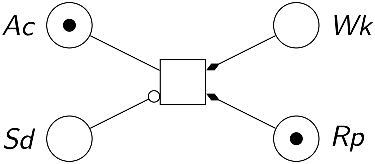
Translation to Petri nets:

- entity $\mapsto$ two complementary places
- rule/constraint $\mapsto$ set of transitions
- transitions priorities
- static elimination of side-loops

Petri nets semantics

with read/inhibitor/reset arcs ⟺ ecosystemic hypergraph

```
Ac+, Sd- >> Wk-, Rp-
```



Remark: can be translated to standard Petri nets (original semantics)

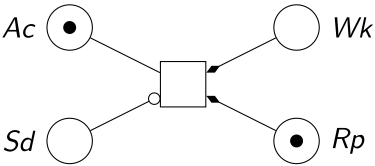
⇒ side-loops, complementary places, several transitions per rule

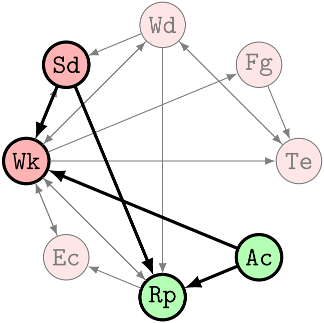
Problem: can we unfold such nets? (master intern wanted)

Petri nets semantics

with read/inhibitor/reset arcs ⟺ ecosystemic hypergraph

`Ac+, Sd- >> Wk-, Rp-`





Remark: can be translated to standard Petri nets (original semantics)
⇒ side-loops, complementary places, several transitions per rule

Problem: can we unfold such nets? (master intern wanted)

Semantics equivalence

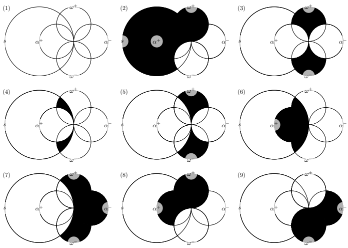
reaction rules —*Petri net semantics*→ Petri net

reaction rules —*state space semantics*→ state space

Petri net —*marking graph semantics*→ marking graph

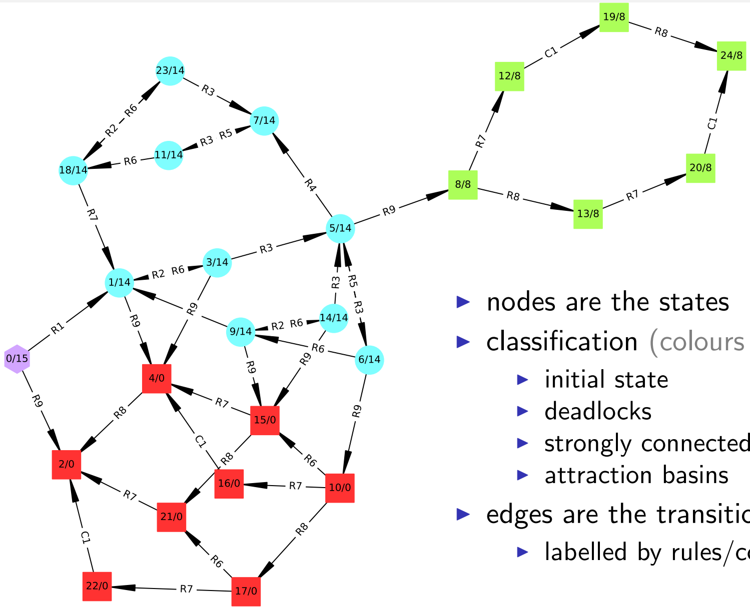
state space ←*equivalence*→ marking graph

(theorem)



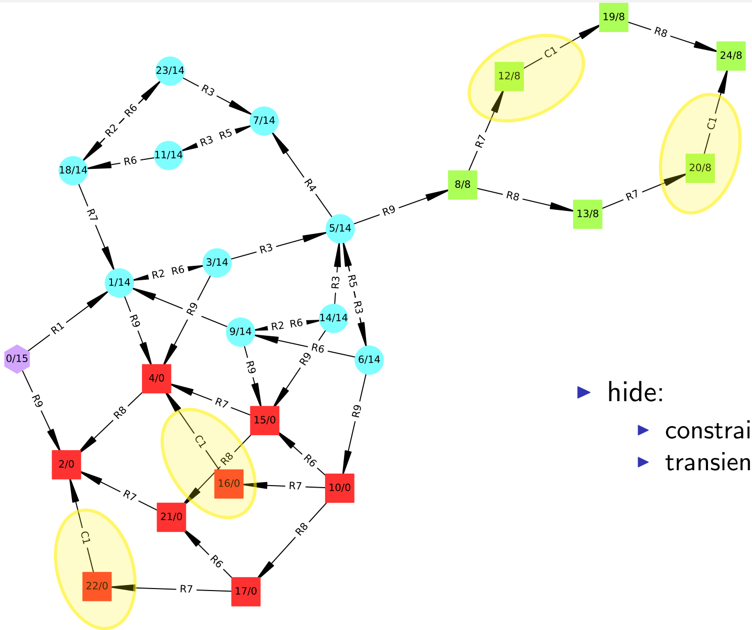
Outline

Full state space



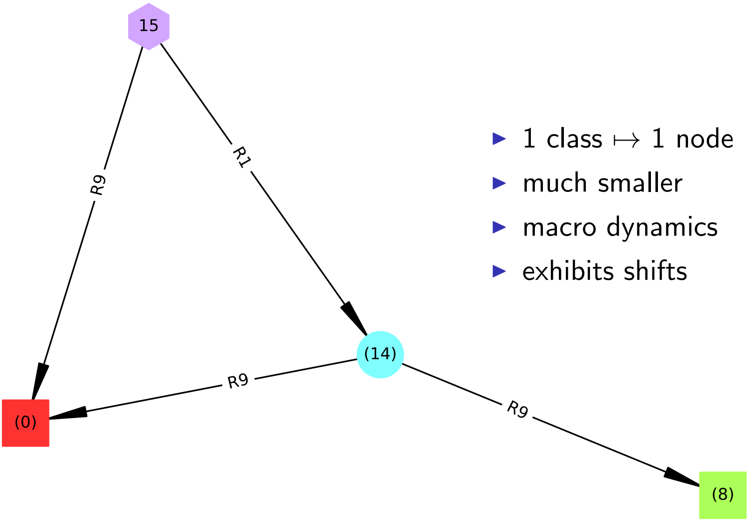
- nodes are the states
- classification (colours & shapes)
  - initial state
  - deadlocks
  - strongly connected components
  - attraction basins
- edges are the transitions
  - labelled by rules/constraints

Compact state spaces



- hide:
  - constraints
  - transient states

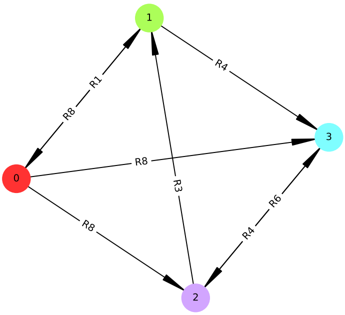
Merged state space



- 1 class ↦ 1 node
- much smaller
- macro dynamics
- exhibits shifts

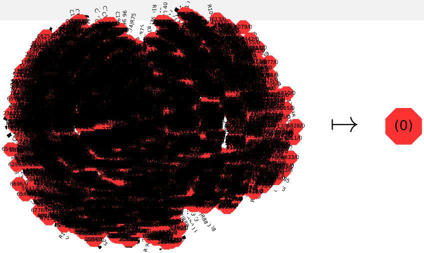
Findings

- clear identification of collapses (deadlocks + basins)
  - direct visualisation of the causes: transitions leading to deadlocks' basins
- direct depiction of catastrophic shifts
  - transitions between components (SCC → SCC/deadlock)
- identification of necessary actions
  - transitions required to reach a SCC
- insights about resilience within SCC
  - distance to exit
  - asymmetric paths (hysteresis)
- identification of the crucial processes
  - transitions that avoid collapses

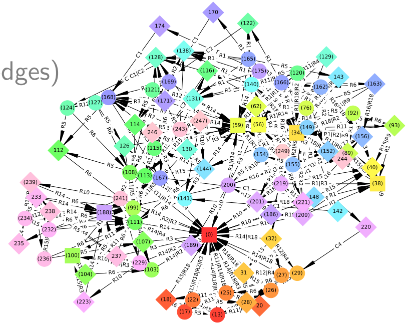


Outline

Size does matter



- degenerated cases
  - a single large SCC
- unfriendly cases
  - two many components
- "just large" cases
  - too many states (4,216,208)
  - too many components (304,646)
  - too many everything (13,214,272 edges)



Mitigation:

- compact state spaces
  - typically: 20 to 50% reduction
- discard small SCC
  - merged into basins
  - small = unimportant?

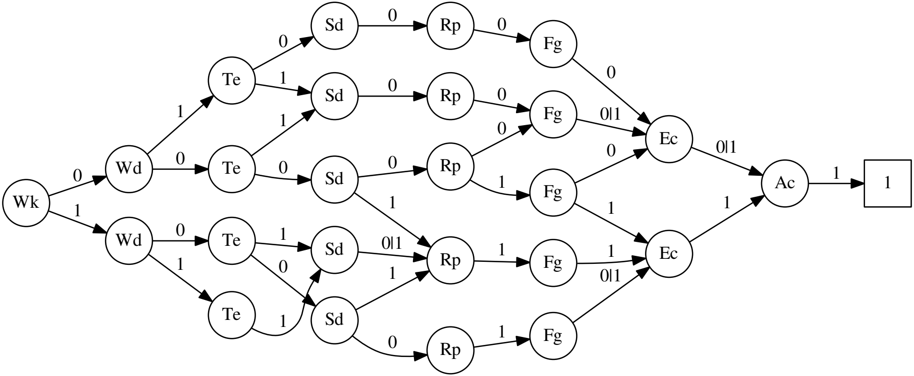
Replacing merges with splits

Explicit approach:

1. explicitly compute states
2. merge classes

Symbolic approach:

1. compute symbolic state space
2. extract/split classes

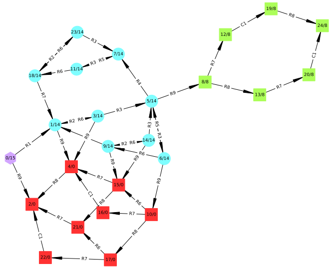


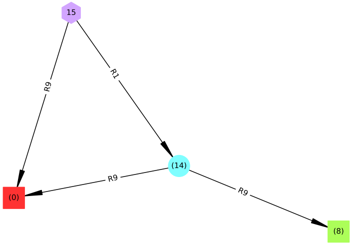
Component graph

Definition

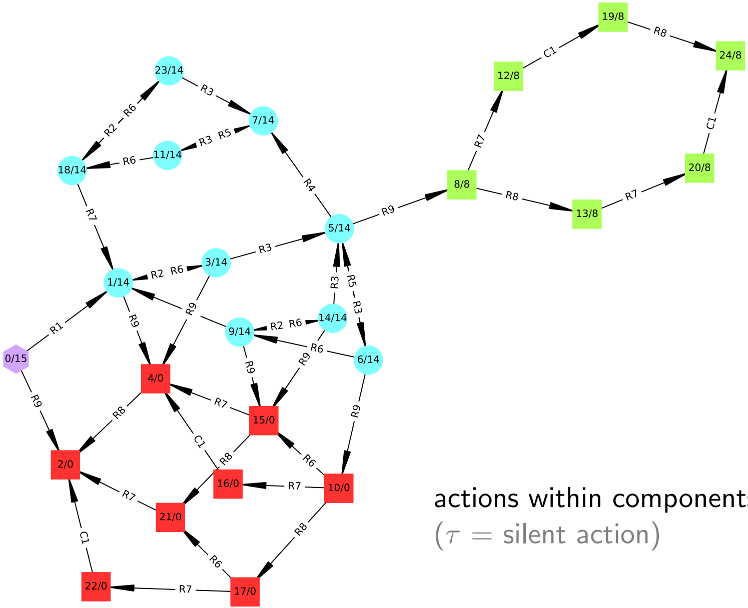
Let $L \stackrel{\text{df}}{=} (\mathcal{R}, s_0, \mathcal{A}, \rightarrow)$ be a labelled transition system (LTS).

1. Component decomposition of L: a partition $\mathcal{C}$ of $\mathcal{R}$.
   - topological components: initial state, deadlocks, SCC, basins
2. Component graph of L wrt $\mathcal{C}$: LTS $L/\mathcal{C} \stackrel{\text{df}}{=} (\mathcal{C}, \langle s_0 \rangle_{\mathcal{C}}, \mathcal{A}_{\mathcal{C}}, \rightarrow_{\mathcal{C}})$ with
   - $\langle s \rangle_{\mathcal{C}} \in \mathcal{C}$ such that $s \in \langle s \rangle_{\mathcal{C}}$
   - $\rightarrow_{\mathcal{C}} \stackrel{\text{df}}{=} \{(\langle s \rangle_{\mathcal{C}}, a, \langle s' \rangle_{\mathcal{C}}) \mid s \xrightarrow{a} s' \wedge \langle s \rangle_{\mathcal{C}} \neq \langle s' \rangle_{\mathcal{C}}\}$
   - $\mathcal{A}_{\mathcal{C}} \stackrel{\text{df}}{=} \{a \in \mathcal{A} \mid \exists C, C' \in \mathcal{C} : C \xrightarrow{a}_{\mathcal{C}} C'\}$





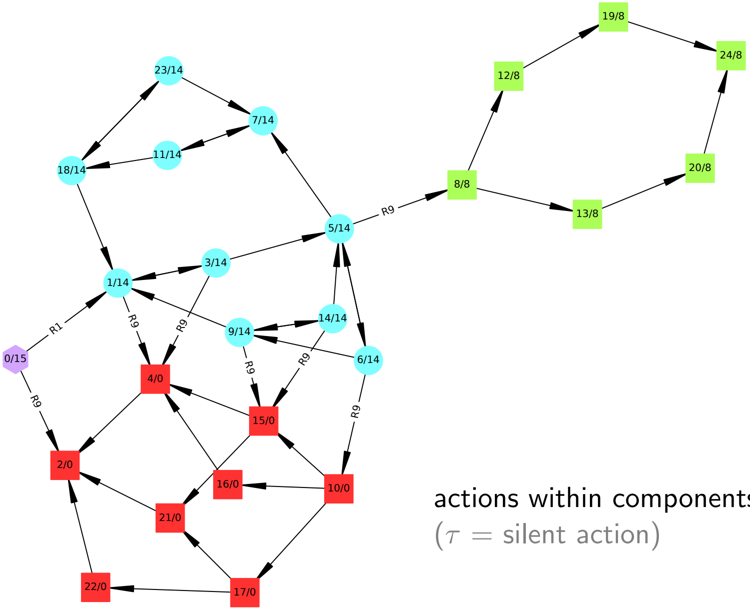
Silencing internal actions



actions within components $\mapsto \tau$

(τ = silent action)

Silencing internal actions



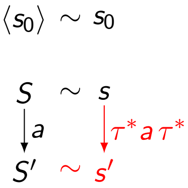
actions within components $\mapsto \tau$

(τ = silent action)

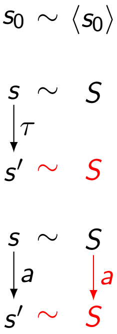
Weak simulations

Correct/complete abstractions

$L/\mathcal{C} \precsim L$



$L \precsim L/\mathcal{C}$



Remark: if both, we have a cosimulation (not a bisimulation)

Proposition: $L \precsim L/\mathcal{C}$ always holds

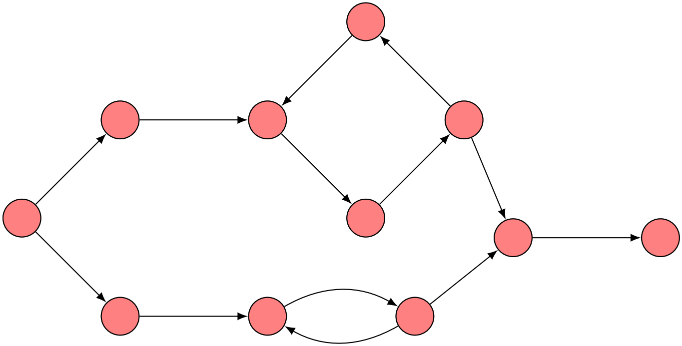
Symbolic primitives

Efficiently computable on decision diagrams

- successor function: $\text{succ}(S) \stackrel{\text{df}}{=} \{s' \mid s \in S \wedge s \rightarrow s'\}$
- predecessor function: $\text{pred}(S) \stackrel{\text{df}}{=} \{s' \mid s \in S \wedge s' \rightarrow s\}$
- identity function: id
- least fixed point of succ: $\text{succ}^* \stackrel{\text{df}}{=} \text{fixpoint}(\text{succ} \cup \text{id})$
- least fixed point of pred: $\text{pred}^* \stackrel{\text{df}}{=} \text{fixpoint}(\text{pred} \cup \text{id})$
- $\text{reach}^* \stackrel{\text{df}}{=} \text{succ}^* \cap \text{pred}^*$
- greatest fixed point of succ: $\text{succ}^\omega \stackrel{\text{df}}{=} \text{fixpoint}(\text{succ} \cap \text{id})$
- greatest fixed point of pred: $\text{pred}^\omega \stackrel{\text{df}}{=} \text{fixpoint}(\text{pred} \cap \text{id})$
- $\text{reach}^\omega \stackrel{\text{df}}{=} \text{succ}^\omega \cap \text{pred}^\omega$

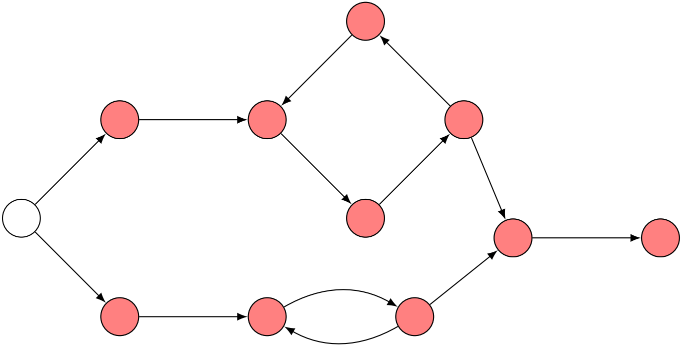
Computing SCC (1/2)

$\text{succ}^\omega \cap \text{pred}^\omega = \text{reach}^\omega$ $\text{succ}^* \cap \text{pred}^* = \text{reach}^*$



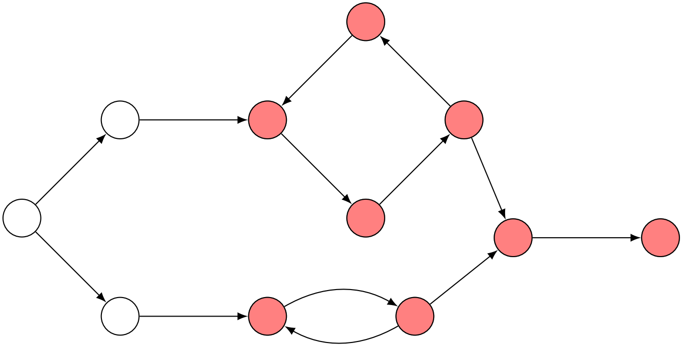
Computing SCC (1/2)

$\text{succ}^\omega \cap \text{pred}^\omega = \text{reach}^\omega$ $\text{succ}^* \cap \text{pred}^* = \text{reach}^*$



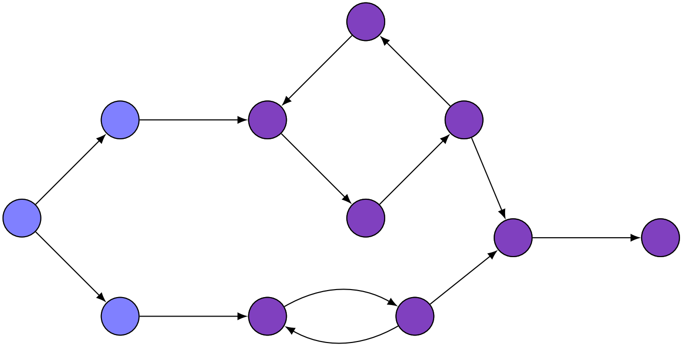
Computing SCC (1/2)

$\mathsf{succ}^\omega \cap \mathsf{pred}^\omega = \mathsf{reach}^\omega$ $\mathsf{succ}^* \cap \mathsf{pred}^* = \mathsf{reach}^*$



Computing SCC (1/2)

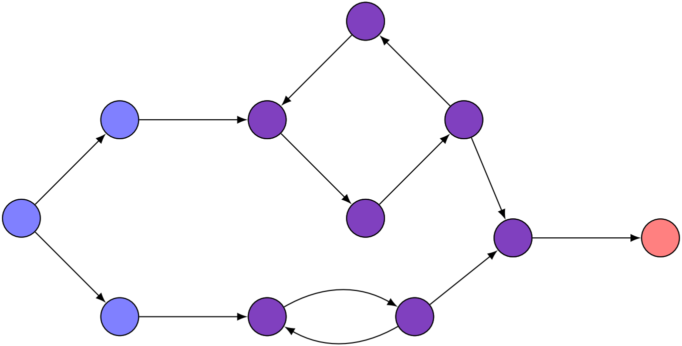
$\text{succ}^\omega \cap \text{pred}^\omega = \text{reach}^\omega$ $\text{succ}^* \cap \text{pred}^* = \text{reach}^*$



Computing SCC (1/2)

$\text{succ}^{\omega} \cap \text{pred}^{\omega} = \text{reach}^{\omega}$

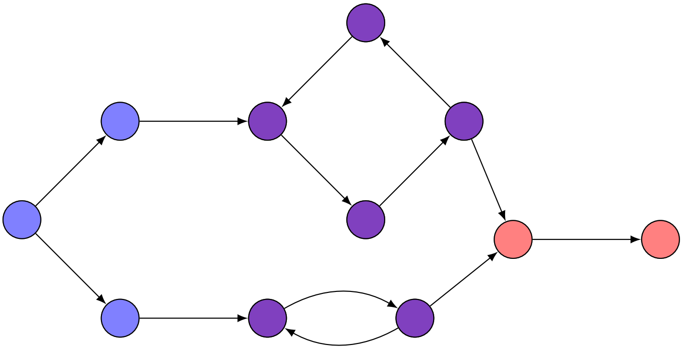
$\text{succ}^{*} \cap \text{pred}^{*} = \text{reach}^{*}$



Computing SCC (1/2)

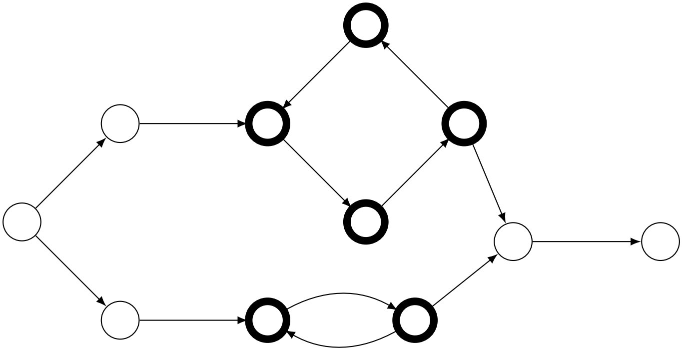
$\text{succ}^{\omega} \cap \text{pred}^{\omega} = \text{reach}^{\omega}$

$\text{succ}^{*} \cap \text{pred}^{*} = \text{reach}^{*}$



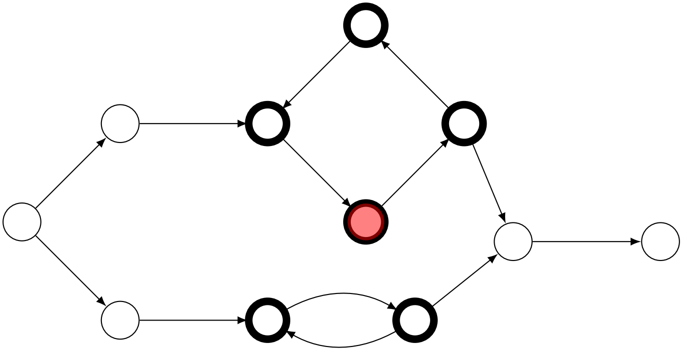
Computing SCC (1/2)

$\text{succ}^\omega \cap \text{pred}^\omega = \text{reach}^\omega$ $\text{succ}^* \cap \text{pred}^* = \text{reach}^*$



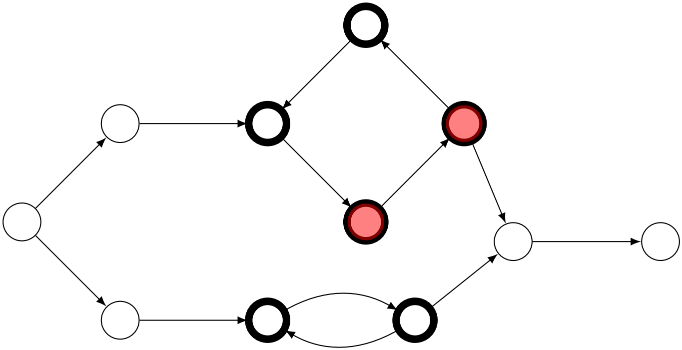
Computing SCC (1/2)

$succ^{\omega} \cap pred^{\omega} = reach^{\omega}$ $succ^{*} \cap pred^{*} = reach^{*}$



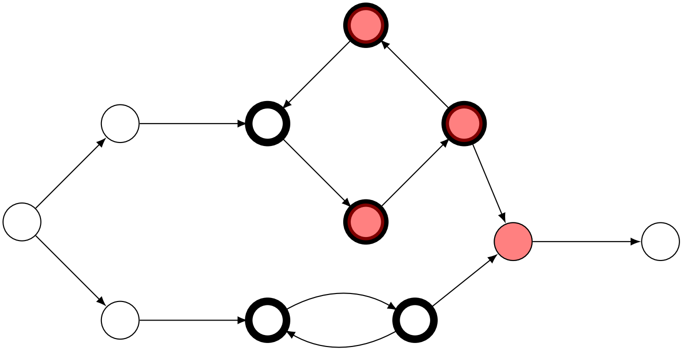
Computing SCC (1/2)

$\text{succ}^{\omega} \cap \text{pred}^{\omega} = \text{reach}^{\omega}$ $\text{succ}^{*} \cap \text{pred}^{*} = \text{reach}^{*}$



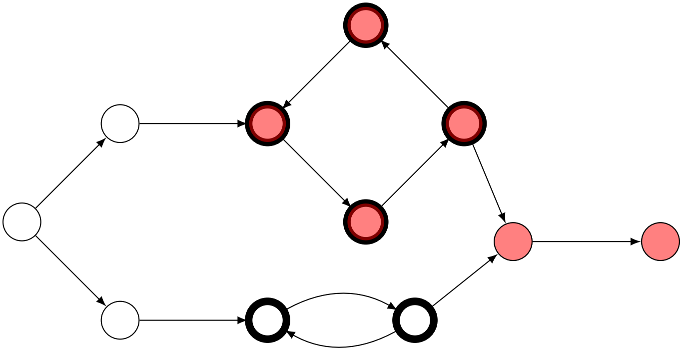
Computing SCC (1/2)

$\mathrm{succ}^\omega \cap \mathrm{pred}^\omega = \mathrm{reach}^\omega$ $\mathrm{succ}^* \cap \mathrm{pred}^* = \mathrm{reach}^*$



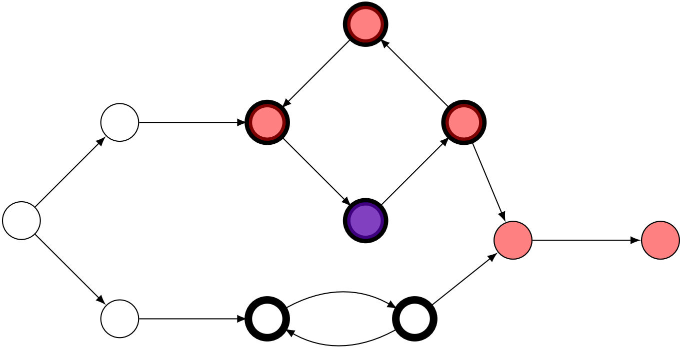
Computing SCC (1/2)

$\text{succ}^\omega \cap \text{pred}^\omega = \text{reach}^\omega$ $\text{succ}^* \cap \text{pred}^* = \text{reach}^*$



Computing SCC (1/2)

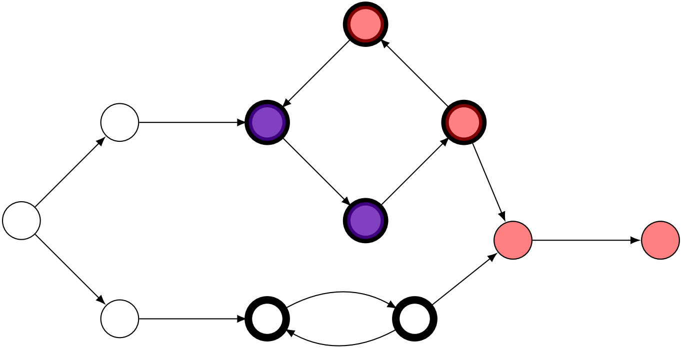
$succ^{\omega} \cap pred^{\omega} = reach^{\omega}$ $succ^{*} \cap pred^{*} = reach^{*}$



Computing SCC (1/2)

$\text{succ}^{\omega} \cap \text{pred}^{\omega} = \text{reach}^{\omega}$

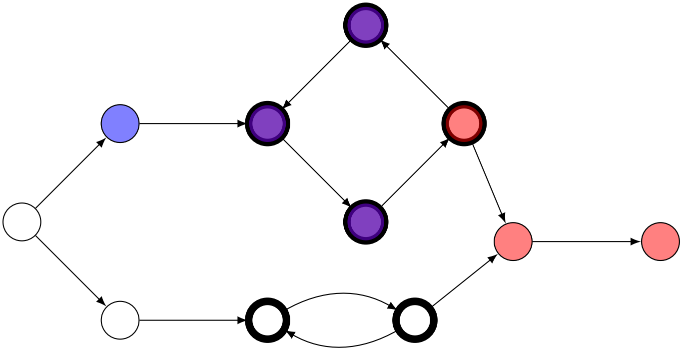
$\text{succ}^{*} \cap \text{pred}^{*} = \text{reach}^{*}$



Computing SCC (1/2)

$\text{succ}^{\omega} \cap \text{pred}^{\omega} = \text{reach}^{\omega}$

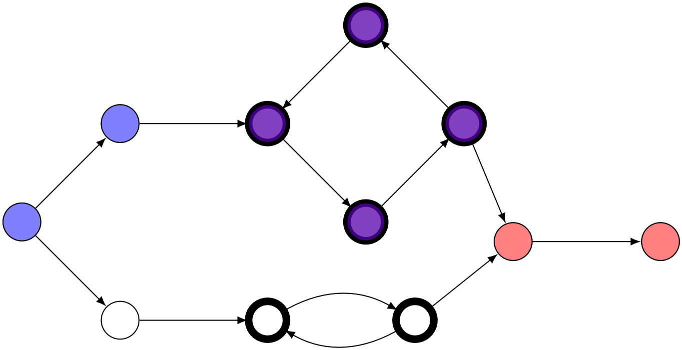
$\text{succ}^{*} \cap \text{pred}^{*} = \text{reach}^{*}$



Computing SCC (1/2)

$\text{succ}^\omega \cap \text{pred}^\omega = \text{reach}^\omega$

$\text{succ}^* \cap \text{pred}^* = \text{reach}^*$



Computing SCC (2/2)

```
def SCC (ℛ) ↦ 𝒮 := ∅:
    H := reach^ω(ℛ)
    while H ≠ ∅:
        pick s ∈ H
        S := reach*(s)
        if |S| > 1:
            𝒮 := 𝒮 ∪ {S}
        H := reach^ω(H \ S)
```

Computing basins

def Basins $(\mathcal{R}, \mathcal{I}) \mapsto \mathcal{B}$**:**

  $\mathcal{D} := \mathcal{R} \setminus \text{pred}(\mathcal{R})$

  $\mathcal{P} := \text{SCC}(\mathcal{R}) \cup \{\{d\} \mid d \in \mathcal{D}\}$

  if $\mathcal{I} \cap C = \emptyset \quad \forall C \in \mathcal{P}$**:**

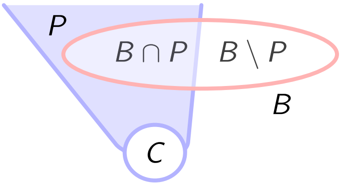
    $\mathcal{P} := \mathcal{P} \cup \{\mathcal{I}\}$

  $\mathcal{B} := \{\mathcal{R} \setminus \cup_{C \in \mathcal{P}} C\}$

  for $C \in \mathcal{P}$**:**

    $P := \text{pred}^*(C)$

    $\mathcal{B} := \{B \cap P, B \setminus P \mid B \in \mathcal{B}\} \setminus \emptyset$



Computing compact graphs

Symbolic computation as previously with:

- constraints ► $\mathcal{U} \stackrel{\text{df}}{=} u_1 \cup \cdots \cup u_k$
- transient states ► $\mathcal{T} \stackrel{\text{df}}{=} \text{pred}_{\mathcal{U}}(\mathcal{R})$
- initial state ► $S_0' \stackrel{\text{df}}{=} \text{succ}_{\mathcal{U}}{}^*(\{s_0\}) \setminus \mathcal{T}$
- successor function ► $\text{succ}' \stackrel{\text{df}}{=} (\text{succ}_{\mathcal{U}}{}^* \circ \text{succ}) \setminus \mathcal{T}$
- predecessor function ► $\text{pred}' \stackrel{\text{df}}{=} (\text{pred} \circ \text{pred}_{\mathcal{U}}{}^*) \setminus \mathcal{T}$

Proposition: compact state space is always bisimilar to full state space

Outline

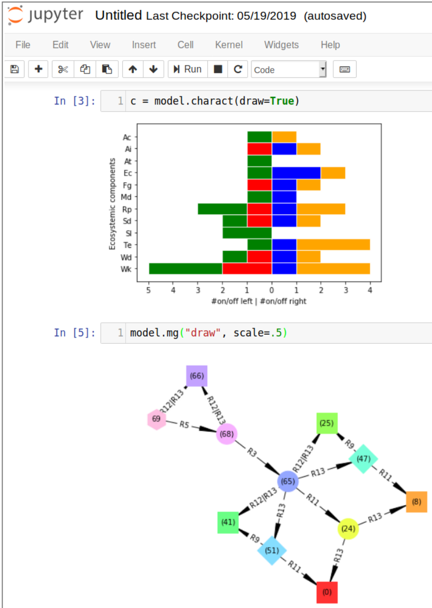
Scientific production & activity

- 1 founding paper
- 1 conference + journal paper
- 3 funded projects
- 12+ master internships
  $\Rightarrow$ more papers in the queue
- 2 PhD in progress
- a software implementation

Methods in Ecology and Evolution BRITISH ECOLOGICAL SOCIETY

Editors: Rob Freckleton, Aaron Ellison, Lee Hsiang Liow and Bob O'Hara

Impact factor: 6.36
ISI Journal Citation Reports © Ranking 2017:
9/158 (Ecology)
Online ISSN: 2041-210X



Ongoing & future works

- coarser-grained decompositions
  - consider deadlocks and SCC hull as a whole
- semi-symbolic state-space
  - compute explicitly the successors of each symbolic state
  - use compiled model (bitfields & bitwise logic)
- user-guided decomposition
  - irreversible transitions
  - measures on the states classes (PCA)
  - Petri nets transitions invariants, unfoldings
  - hints & requests from the modeller (LTL/CTL)
- other trends of research (with H Klaudel & C. Di Giusto)
  - quantitative modelling with simulation
  - static model reductions
  - patterns identification
  - comparisons of models