IMT SCHOOL FOR ADVANCED STUDIES LUCCA

QUANTIVATIVE ABSTRACTIONS FOR COLLECTIVE ADAPTIVE SYSTEMS

Mirco Tribastone and Andrea Vandin

Based on joint work with
Luca Cardelli and Max Tschaikowski

SFM'16 - Bertinoro

24 June 2016

Collective Adaptive Systems

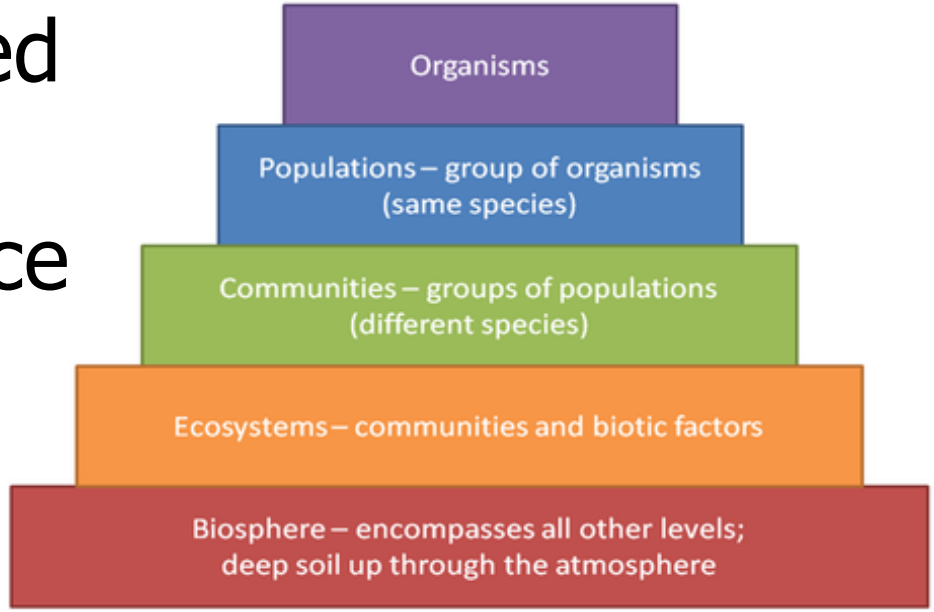
- CAS are collections of entities interacting with each other and the environment in such a way that the overall behavior cannot be understood by analyzing each agent in isolation
- Modeling CAS raises challenges:
  - The overall behavior requires building the "product space" of the individual state spaces
  - **State explosion** makes the analysis unfeasible in real-world situations

Quantitative abstractions

- Focus on **quantitative** properties of CAS
- Standard techniques for quantitative analysis do not scale with large populations of individuals
  (see e.g. the lectures of Nicolas/Luca and Jane/Michele)
- **Abstractions** are more compact representations that preserve some of the original behavior
  (see e.g. Vashti's lecture for spatial abstractions)

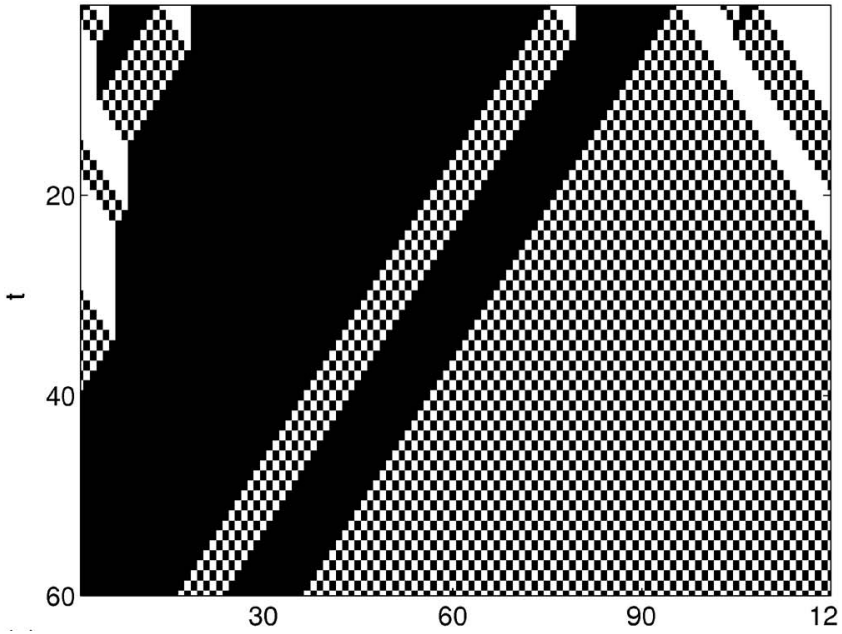
Motivating examples

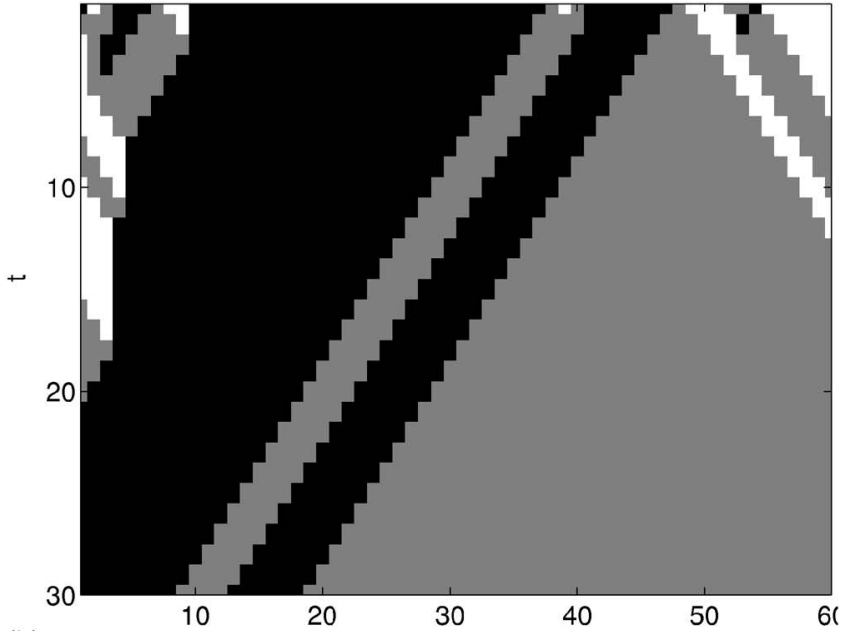
- **Ecology** is a prime domain of (natural) CAS
  - Populations of individuals interact with each other and adapt to the external environment
  - Different kinds of individuals may have conflicting objectives (e.g., predators vs preys)
- Abstractions are needed to cope with multiple scales in time and space



Motivating examples

- In physics, cellular automata are a basic model based on interaction rules between simple neighboring agents (see Vashti's lecture)
- Abstraction recover overall dynamics ignoring low-level details

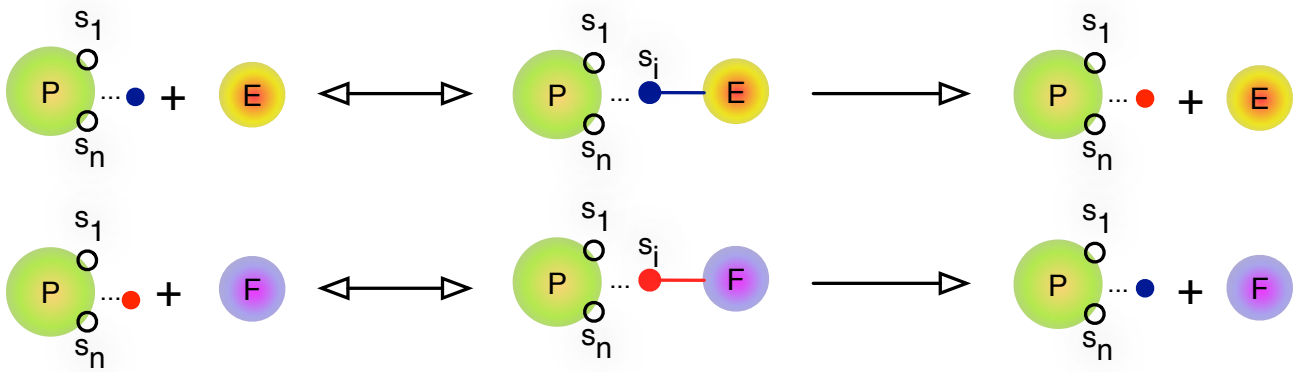




Israeli, Godenfeld. Phys. Rev. E **73**, 026203 2006

Motivating examples

- In systems biology, combinatorial explosion arises from the mechanistic modeling of protein interaction networks



$$P(b,b,\ldots,b) + E \xrightarrow{k_1} P(E,b,\ldots,b) \qquad P(r,b,\ldots,b) + F \xrightarrow{k_2} P(F,b,\ldots,b)$$

$$P(b,b,\ldots,b) + E \xrightarrow{k_1} P(b,E,\ldots,b) \qquad P(b,r,\ldots,b) + F \xrightarrow{k_2} P(b,F,\ldots,b)$$

$$\ldots \qquad \ldots$$

$$P(b,b,\ldots,b) + E \xrightarrow{k_1} P(b,b,\ldots,E) \qquad P(b,b,\ldots,r) + F \xrightarrow{k_2} P(b,b,\ldots,F)$$

$$\ldots \qquad \ldots$$

Quantitative analysis of CAS

- Ordinary differential equations (ODEs) are popular in CAS modeling:
  - Ecology, epidemiology: populations of individuals
  - Systems biology: concentrations of complexes
  - Control engineering: pressure, temperature,…
  - Computer science:
    - Transient probability distribution of a Markov chain is described by a (large) linear ODE system
    - **Fluid approximations** provide more compact (typically non-linear) ODE models of Markov population processes [Hillston'05]

Lecture overview

1. Brief intro to ODEs
2. Abstraction through **ODE reduction**
   (orthogonal to other approaches that interpolate ODEs as PDEs for spatial limits, see e.g. Vashti's lecture)
   - Symbolic differential equivalences
   - (Structural) bisimulations for reaction networks
3. Perspectives and open challenges
4. Case studies with **ERODE** (A. Vandin)

ODEs: minimal introduction

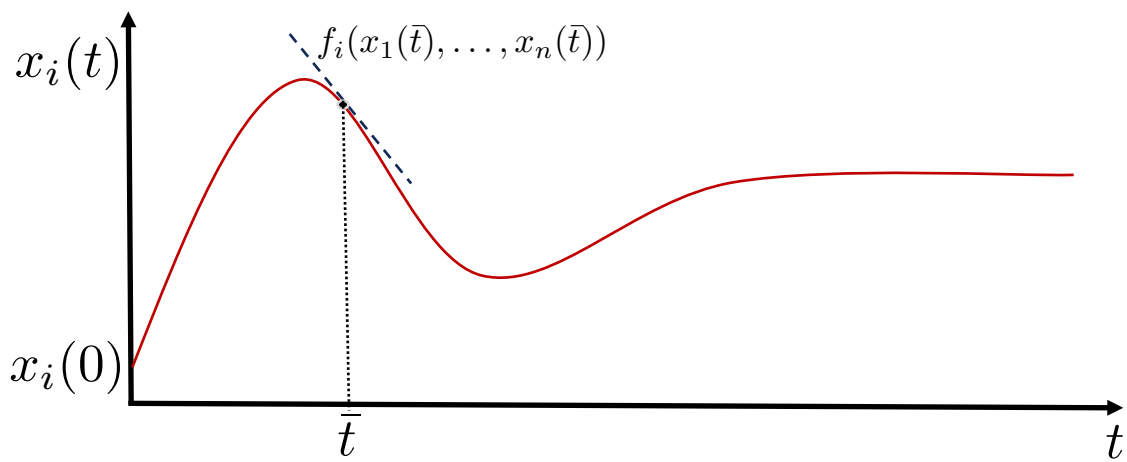
First-order explicit system of ordinary differential equations

$$\frac{dx_1(t)}{dt} = f_1(x_1(t), \ldots, x_n(t))$$
$$\frac{dx_2(t)}{dt} = f_2(x_1(t), \ldots, x_n(t))$$
$$\ldots$$
$$\frac{dx_n(t)}{dt} = f_n(x_1(t), \ldots, x_n(t))$$

also written

$$\dot{x}_1 = f_1(x)$$
$$\dot{x}_2 = f_2(x)$$
$$\ldots$$
$$\dot{x}_n = f_n(x)$$

Initial value problem: Find the trajectories $x_i(t)$ that satisfy the ODEs when starting from $x_i(0)$



ODEs: minimal introduction

First-order explicit system of ordinary differential equations

$$\frac{dx_1(t)}{dt} = f_1(x_1(t), \ldots, x_n(t))$$
$$\frac{dx_2(t)}{dt} = f_2(x_1(t), \ldots, x_n(t))$$
$$\ldots$$
$$\frac{dx_n(t)}{dt} = f_n(x_1(t), \ldots, x_n(t))$$

also written

$$\dot{x}_1 = f_1(x)$$
$$\dot{x}_2 = f_2(x)$$
$$\ldots$$
$$\dot{x}_n = f_n(x)$$

Solution (exists and is unique in our models):

$$\frac{dx_i(t)}{dt} \approx \frac{x_i(t+\Delta t) - x_i(t)}{\Delta t}$$
$$x_i(t+\Delta t) \approx x_i(t) + \Delta t \cdot f_i(x_1(t), \ldots, x_n(t))$$

Problem statement

First-order explicit system of ordinary differential equations

$$\frac{dx_1(t)}{dt} = f_1(x_1(t), \ldots, x_n(t))$$
$$\frac{dx_2(t)}{dt} = f_2(x_1(t), \ldots, x_n(t))$$
$$\ldots$$
$$\frac{dx_n(t)}{dt} = f_n(x_1(t), \ldots, x_n(t))$$

also written

$$\dot{x}_1 = f_1(x)$$
$$\dot{x}_2 = f_2(x)$$
$$\ldots$$
$$\dot{x}_n = f_n(x)$$

$$x_i(t + \Delta t) \approx x_i(t) + \Delta t \cdot f_i(x_1(t), \ldots, x_n(t))$$

- In many cases, *n* can be very large (e.g., order of millions)
- Numerical solution may become computationally prohibitive

Abstraction through reduction

A lower dimensional ODE that preserves "some" of the original dynamics

$$\dot{x}_1 = f_1(x)$$
$$\dot{x}_2 = f_2(x)$$
$$\dots$$
$$\dot{x}_n = f_n(x)$$

$$m \ll n$$

Nonlinear ODEs
Automatic
Scalable
Exact

$$\dot{y}_1 = g_1(y)$$
$$\dots$$
$$\dot{y}_m = g_m(y)$$

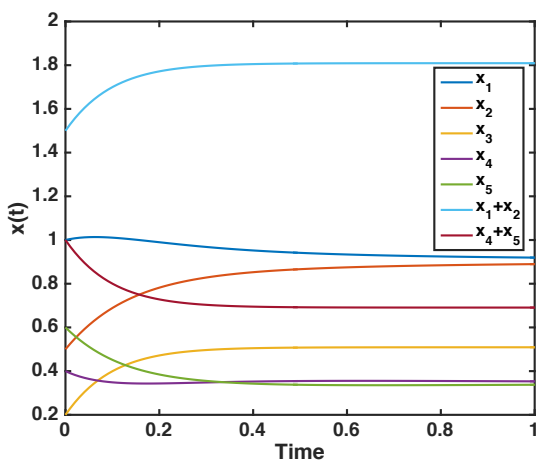
Forward equivalence

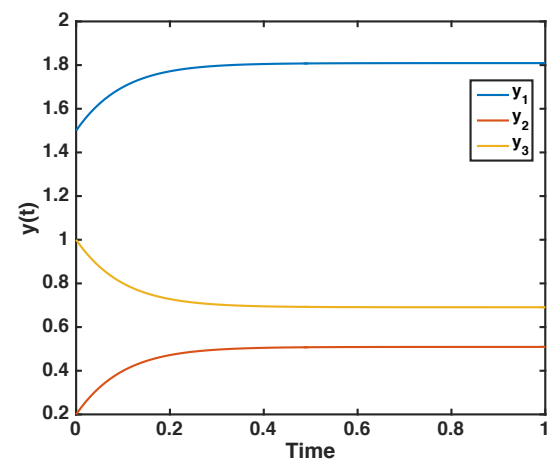
A partition of the variables yielding an equation for each block (sum of variables)

$$\dot{x}_1 = -x_1 + x_2 - 3x_1x_3 + 4x_4$$
$$\dot{x}_2 = +x_1 - x_2 - 3x_2x_3 + 4x_5$$
$$\dot{x}_3 = -3x_1x_3 + 4x_4 - 3x_2x_3 + 4x_5$$
$$\dot{x}_4 = +3x_1x_3 - 4x_4$$
$$\dot{x}_5 = +3x_2x_3 - 4x_5$$

$$\overset{y_1}{\updownarrow} \quad \overset{y_2}{\updownarrow} \quad \overset{y_3}{\updownarrow}$$
$$\{\{x_1, x_2\}, \{x_3\}, \{x_4, x_5\}\}$$

$$\dot{y}_1 = -3y_1y_2 + 4y_3$$
$$\dot{y}_2 = -3y_1y_2 + 4y_3$$
$$\dot{y}_3 = +3y_1y_2 - 4y_3$$





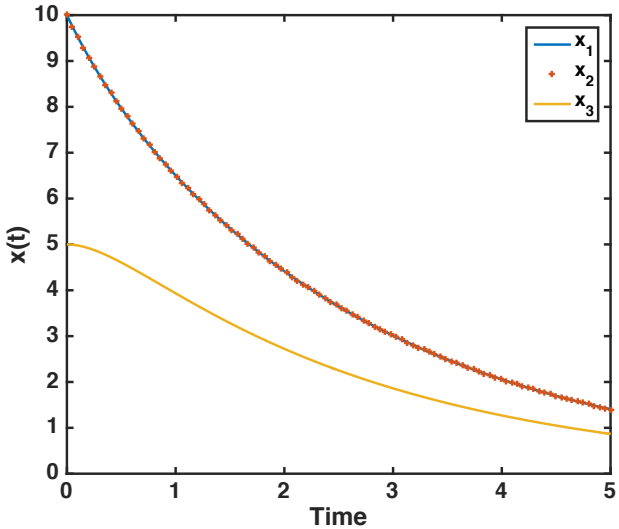
Backward equivalence

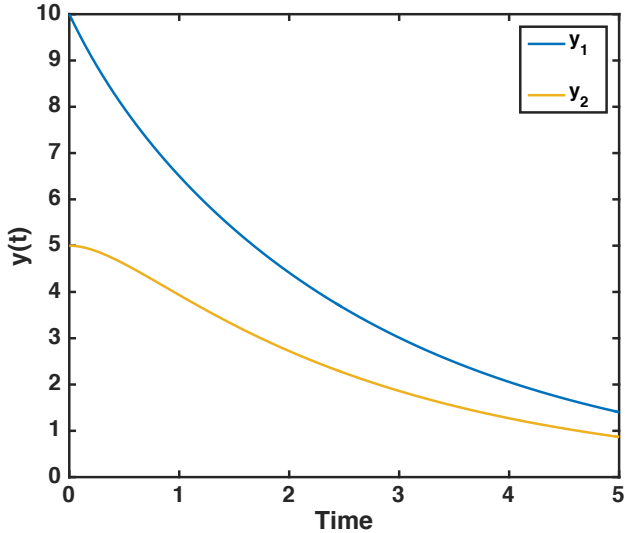
Equivalent variables have the same solutions at all time points

$$\dot{x}_1 = -\min(x_1, x_2) + x_3$$
$$\dot{x}_2 = -\min(x_1, x_2) + x_3$$
$$\dot{x}_3 = +\min(x_1, x_2) - 2x_3$$

$$\{\{\overset{y_1}{x_1}, x_2\}, \{\overset{y_2}{x_3}\}\}$$

$$\dot{y}_1 = -y_1 + y_2$$
$$\dot{y}_2 = +y_1 - 2y_2$$





Language-driven reasoning

IDOL: Intermediate Drift-oriented Language **[POPL'16]**

$$p ::= \varepsilon \mid \dot{x}_i = f,\ p$$

$$f ::= n \mid x_i \mid f + f \mid f \cdot f \mid f^{\frac{1}{m}}$$

- ODEs covered:
  - Polynomials of any degree
  - Rational expressions
  - Minima/maxima
  - …
- Forward/backward equivalences are **fully characterized**
- Symbolic partition refinement via an **SMT encoding**

Reaction Networks **[CONCUR'15, TACAS'16]**

$$x_i + x_j \xrightarrow{k_1} x_i + x_j + x_k$$

$$x_i \xrightarrow{k_2} x_i + x_j$$

$$\emptyset \xrightarrow{k_3} x_i$$

- Multivariate polynomials of degree **at most two**
- Bisimulations over the "species" of the reaction network
- Forward bisimulation is a **sufficient condition** only
- Backward bisimulation fully characterized
- **Polynomial** partition refinement

Language-driven reasoning

IDOL: Intermediate Drift-oriented Language [POPL'16]

$$p ::= \varepsilon \mid \dot{x}_i = f,\ p$$

$$f ::= n \mid x_i \mid f + f \mid f \cdot f \mid f^{\frac{1}{m}}$$

Reaction Networks [CONCUR'15, TACAS'16]

$$x_i + x_j \xrightarrow{k_1} x_i + x_j + x_k$$

$$x_i \xrightarrow{k_2} x_i + x_j$$

$$\emptyset \xrightarrow{k_3} x_i$$

Scalability

Expressiveness/Completeness

Language-driven reasoning

IDOL: Intermediate Drift-oriented Language **[POPL'16]**

$$p ::= \varepsilon \mid \dot{x}_i = f,\ p$$

$$f ::= n \mid x_i \mid f + f \mid f \cdot f \mid f^{\frac{1}{m}}$$

Reaction Networks

[CONCUR'15, TACAS'16]

$$x_i + x_j \xrightarrow{k_1} x_i + x_j + x_k$$

$$x_i \xrightarrow{k_2} x_i + x_j$$

$$\emptyset \xrightarrow{k_3} x_i$$

This lecture:

- Forward equivalence
- **Backward equivalence**

- **Forward bisimulation**
- Backward bisimulation

IDOL backward equivalence

- Basic idea:
  - An assignment $(x_1, \ldots, x_n)$ is **uniform** on a partition of variables if it has equal values for equivalent variables.
    - $(x_1, x_2, x_3) = (1, 1, 0)$ is uniform on $\{\{x_1, x_2\}, \{x_3\}\}$
  - **Theorem:** A partition of variables is a backward equivalence iff for any uniform assignment its derivative is also uniform
  - Encode this condition in first-order logic!

Backward equivalence: example

Model

$$\dot{x}_1 = -\min(x_1, x_2) + x_3$$
$$\dot{x}_2 = -\min(x_1, x_2) + x_3$$
$$\dot{x}_3 = +\min(x_1, x_2) - 2x_3$$

Candidate partition

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

Equivalence condition (quantifier free)

$$\phi := (x_1 = x_2) \implies -\min(x_1, x_2) + x_3 = -\min(x_1, x_2) + x_3$$

SMT check

$$\mathrm{sat}(\neg\phi) = \mathbf{false}$$

IDOL partition refinement

Algorithm. Compute the largest equivalence that refines a given partition of variables.

1. SMT check
2. If **sat** get witness and split partition preserving its uniformity. Goto 1.
3. If **unsat** the current partition is the coarsest refinement. End.

$$\dot{x}_1 = -\min(x_1, x_2) + x_3$$
$$\dot{x}_2 = -\min(x_1, x_2) + x_3$$
$$\dot{x}_3 = +\min(x_1, x_2) - 2x_3$$

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

$$f((1,1,1)) = (0,0,-1)$$

sat

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

unsat

IDOL partition refinement

Algorithm. Compute the largest equivalence that refines **a given partition of variables**.

$$\dot{x}_1 = -\min(x_1, x_2) + x_3$$
$$\dot{x}_2 = -\min(x_1, x_2) + x_3$$
$$\dot{x}_3 = +\min(x_1, x_2) - 2x_3$$

The freedom in choosing the initial partition is useful:

- The largest equivalence can be obtained by initializing the algorithm with the trivial singleton partition
- Other partitions may be used to keep variables distinct (e.g., if they are known to start from different initial conditions)

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

$$f((1,1,1)) = (0,0,-1)$$

sat

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

unsat

Language-driven reasoning

IDOL: Intermediate Drift-oriented Language **[POPL'16]**

$$p ::= \varepsilon \mid \dot{x}_i = f,\ p$$

$$f ::= n \mid x_i \mid f + f \mid f \cdot f \mid f^{\frac{1}{m}}$$

Reaction Networks **[CONCUR'15, TACAS'16]**

$$x_i + x_j \xrightarrow{k_1} x_i + x_j + x_k$$

$$x_i \xrightarrow{k_2} x_i + x_j$$

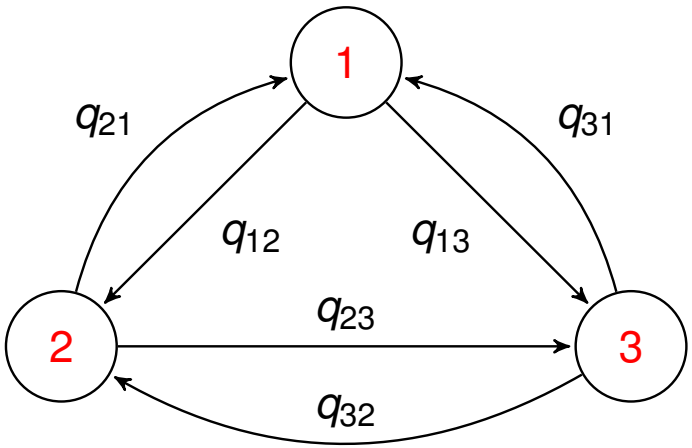
$$\emptyset \xrightarrow{k_3} x_i$$

- Forward equivalence
- Backward equivalence

- **Forward bisimulation**
- Backward bisimulation

Analogy: Markov chain lumping

Transition diagram



Transition matrix

$$Q = \begin{bmatrix} -(q_{12}+q_{13}) & q_{12} & q_{13} \\ q_{21} & -(q_{21}+q_{23}) & q_{23} \\ q_{31} & q_{32} & -(q_{31}+q_{32}) \end{bmatrix}$$

Equations of motion (linear ODE system)

$$\frac{d\pi_i(t)}{dt} = q_{ii}\pi_i(t) + \sum_{j\neq i} q_{ji}\pi_j(t)$$

flux out

flux in

Ordinary lumpability

- Set $Z = \{1, \ldots, n\}$ the state space of a CTMC with matrix $Q = (q_{ij})_{1 \leq i,j \leq n}$
- A partition $X = \{X_1, \ldots, X_N\}$ of Z is **ordinarily lumpable** if for any pair of blocks I, J and any two states $i_1, i_2 \in I$

$$q_{i_1 J} = q_{i_2 J}, \qquad q_{iJ} := \sum_{j \in X_J} q_{ij}$$

- These common values form the **lumped CTMC**

Ordinary lumpability

A special case of forward equivalence:

- Let $\pi(t) = (\pi_1(t), \ldots, \pi_n(t))$ be the solution of the original CTMC
- Let $\hat{\pi}(t) = (\hat{\pi}_1(t), \ldots \hat{\pi}_N(t))$ be the solution of the lumped CTMC
- Set $\Pi_{X_i}(t) := \sum_{j \in X_i} \pi_j(t)$
- Then we have

$$\hat{\pi}_i(0) = \Pi_{X_i}(0) \Longrightarrow \hat{\pi}_i(t) = \Pi_{X_i}(t)$$

[Buchholz'94]

Example

Idle

Think time

$$Q = \begin{array}{c} \\ (I,I) \\ (I,W) \\ (W,I) \\ (W,W) \end{array} \begin{array}{c} \begin{array}{cccc} (I,I) & (I,W) & (W,I) & (W,W) \end{array} \\ \begin{bmatrix} -2\lambda & \lambda & \lambda & 0 \\ \mu & -(\mu+\lambda) & 0 & \lambda \\ \mu & 0 & -(\mu+\lambda) & \lambda \\ 0 & \mu/2 & \mu/2 & -\mu \end{bmatrix} \end{array}$$

Working

Service

$$Q_{\mathcal{N}} = \begin{array}{c} \\ (2,0) \\ (1,1) \\ (0,2) \end{array} \begin{array}{c} \begin{array}{ccc} (2,0) & (1,1) & (0,2) \end{array} \\ \begin{bmatrix} -2\lambda & 2\lambda & 0 \\ \mu & -(\mu+\lambda) & \lambda \\ 0 & \mu & -\mu \end{bmatrix} \end{array}$$

(Symmetry reduction for Markov chains)

Ordinary lumpability

- The criterion

$$q_{i_1 J} = q_{i_2 J}, \qquad q_{iJ} := \sum_{j \in X_J} q_{ij}$$

is **structural** but it implies aggregation at the ODE level

- Idea: Can we find an analogous structural criterion for a more general class of ODE systems?

ODEs via reaction networks

- Defined for reaction networks (RN) with at most two reagents in each reaction:

$$\dot{V}_k = \sum_{1 \leq i,j \leq n} \boxed{\alpha_{i,j}^{(k)} \cdot V_i \cdot V_j} + \sum_{1 \leq i \leq n} \boxed{\alpha_i^{(k)} \cdot V_i} + \boxed{\beta^{(k)}}$$

through mass-action semantics

$$R = \left\{ \mathrm{X}_i + \mathrm{X}_j \xrightarrow{\alpha_{i,j}^{(k)}} \mathrm{X}_i + \mathrm{X}_j + \mathrm{X}_k \mid \alpha_{i,j}^{(k)} \neq 0 \right\} \cup$$
$$\left\{ \mathrm{X}_i \xrightarrow{\alpha_i^{(k)}} \mathrm{X}_i + \mathrm{X}_k \mid \alpha_i^{(k)} \neq 0 \right\} \cup$$
$$\left\{ \emptyset \xrightarrow{\beta^{(k)}} \mathrm{X}_k \mid \beta^{(k)} \neq 0 \right\}$$

Forward bisimulation

Based on quantities from the **RN syntax**

$$\mathbf{crr}[X,\rho] := (\rho(X)+1) \sum_{X+\rho \xrightarrow{\alpha} \pi \in R} \alpha, \quad \mathbf{pr}(X,Y,\rho) := (\rho(X)+1) \sum_{X+\rho \xrightarrow{\alpha} \pi \in R} \alpha \cdot \pi(Y)$$

$$\mathbf{pr}[X,H,\rho] := \textstyle\sum_{Y \in H} \mathbf{pr}(X,Y,\rho)$$

For X, Y in a block, for all blocks H and partners ρ:

$$\mathbf{crr}[X,\rho] = \mathbf{crr}[Y,\rho] \quad \textit{and} \quad \mathbf{pr}[X,H,\rho] = \mathbf{pr}[Y,H,\rho]$$

★ It corresponds to ordinary lumpability for RNs that represent a Markov chain

Bisimulations: algorithm

Initial partition

$$\{\{X_1, X_2, X_3, X_4, X_5\}\}$$

After pre-partitioning for **crr**

$$\{\{X_1, X_2, X_4, X_5\}, \{X_3\}\}$$

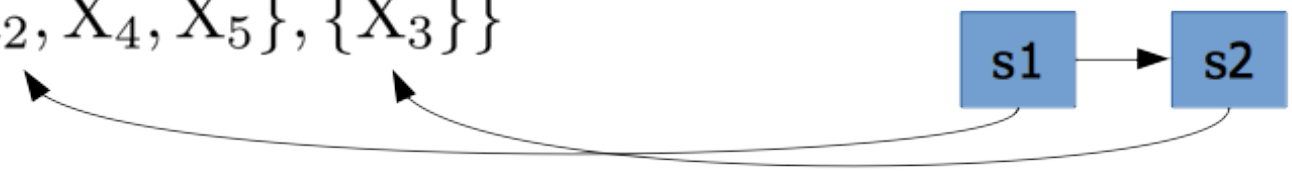
$$\begin{array}{ll}
\mathbf{crr}[X_1, \emptyset] = 1 & \mathbf{crr}[X_1, X_3] = 3 \\
\mathbf{crr}[X_2, \emptyset] = 1 & \mathbf{crr}[X_2, X_3] = 3 \\
\mathbf{crr}[X_3, \emptyset] = 0 & \mathbf{crr}[X_3, X_3] = 0 \\
\mathbf{crr}[X_4, \emptyset] = 1 & \mathbf{crr}[X_4, X_3] = 3 \\
\mathbf{crr}[X_5, \emptyset] = 1 & \mathbf{crr}[X_5, X_3] = 3
\end{array}$$

$$\begin{array}{l}
X_1 \xrightarrow{1} X_2 \\
X_2 \xrightarrow{1} X_1 \\
X_4 \xrightarrow{1} 2X_1 + X_3 \\
X_5 \xrightarrow{1} 2X_2 + X_3 \\
X_1 + X_3 \xrightarrow{3} X_4 \\
X_2 + X_3 \xrightarrow{3} X_5 \\
X_4 + X_3 \xrightarrow{3} X_5 \\
X_5 + X_3 \xrightarrow{3} X_4
\end{array}$$

First iteration

Current partition:

$$\{\{X_1, X_2, X_4, X_5\}, \{X_3\}\}$$



Partitioning according to s1 we get:

$$\{\{X_3\}, \{X_1, X_2\}, \{X_4, X_5\}\}$$

$$\mathbf{pr}[X_1, s1, \emptyset] = 1$$
$$\mathbf{pr}[X_2, s1, \emptyset] = 1$$
$$\mathbf{pr}[X_4, s1, \emptyset] = 2$$
$$\mathbf{pr}[X_5, s1, \emptyset] = 2$$

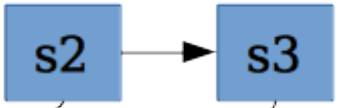
$$X_1 \xrightarrow{1} X_2$$
$$X_2 \xrightarrow{1} X_1$$
$$X_4 \xrightarrow{1} 2X_1 + X_3$$
$$X_5 \xrightarrow{1} 2X_2 + X_3$$
$$X_1 + X_3 \xrightarrow{3} X_4$$
$$X_2 + X_3 \xrightarrow{3} X_5$$
$$X_4 + X_3 \xrightarrow{3} X_5$$
$$X_5 + X_3 \xrightarrow{3} X_4$$

Second iteration

Current partition:

$$\{\{X_3\},\{X_1,X_2\},\{X_4,X_5\}\}$$

Candidate splitters:



No refinement is obtained for s2

$$\mathbf{pr}[X_1, s2, \emptyset] = 0 \quad \mathbf{pr}[X_1, s2, X_3] = 0$$
$$\mathbf{pr}[X_2, s2, \emptyset] = 0 \quad \mathbf{pr}[X_2, s2, X_3] = 0$$

$$\mathbf{pr}[X_4, s2, \emptyset] = 1 \quad \mathbf{pr}[X_4, s2, X_3] = 0$$
$$\mathbf{pr}[X_5, s2, \emptyset] = 1 \quad \mathbf{pr}[X_5, s2, X_3] = 0$$

$$X_1 \xrightarrow{1} X_2$$
$$X_2 \xrightarrow{1} X_1$$
$$X_4 \xrightarrow{1} 2X_1 + X_3$$
$$X_5 \xrightarrow{1} 2X_2 + X_3$$
$$X_1 + X_3 \xrightarrow{3} X_4$$
$$X_2 + X_3 \xrightarrow{3} X_5$$
$$X_4 + X_3 \xrightarrow{3} X_5$$
$$X_5 + X_3 \xrightarrow{3} X_4$$

Third iteration

Current partition:

$$\{\{X_3\},\{X_1,X_2\},\{X_4,X_5\}\}$$

Candidate splitters:

s3

No refinement is obtained for s3

$$\mathbf{pr}[X_1, s3, \emptyset] = 0 \quad \mathbf{pr}[X_1, s3, X_3] = 3$$
$$\mathbf{pr}[X_2, s3, \emptyset] = 0 \quad \mathbf{pr}[X_2, s3, X_3] = 3$$

$$\mathbf{pr}[X_4, s3, \emptyset] = 0 \quad \mathbf{pr}[X_4, s3, X_3] = 3$$
$$\mathbf{pr}[X_5, s3, \emptyset] = 0 \quad \mathbf{pr}[X_5, s3, X_3] = 3$$

Time Complexity: $O(mn \log n)$

$$X_1 \xrightarrow{1} X_2$$
$$X_2 \xrightarrow{1} X_1$$
$$X_4 \xrightarrow{1} 2X_1 + X_3$$
$$X_5 \xrightarrow{1} 2X_2 + X_3$$
$$X_1 + X_3 \xrightarrow{3} X_4$$
$$X_2 + X_3 \xrightarrow{3} X_5$$
$$X_4 + X_3 \xrightarrow{3} X_5$$
$$X_5 + X_3 \xrightarrow{3} X_4$$

Third iteration

Current partition:

$$\{\{X_3\}, \{X_1, X_2\}, \{X_4, X_5\}\}$$

Candidate splitters:

s3

No refinement is obtained for s3

$$\mathbf{pr}[X_1, s3, \emptyset] = 0 \quad \mathbf{pr}[X_1, s3, X_3] = 3$$
$$\mathbf{pr}[X_2, s3, \emptyset] = 0 \quad \mathbf{pr}[X_2, s3, X_3] = 3$$

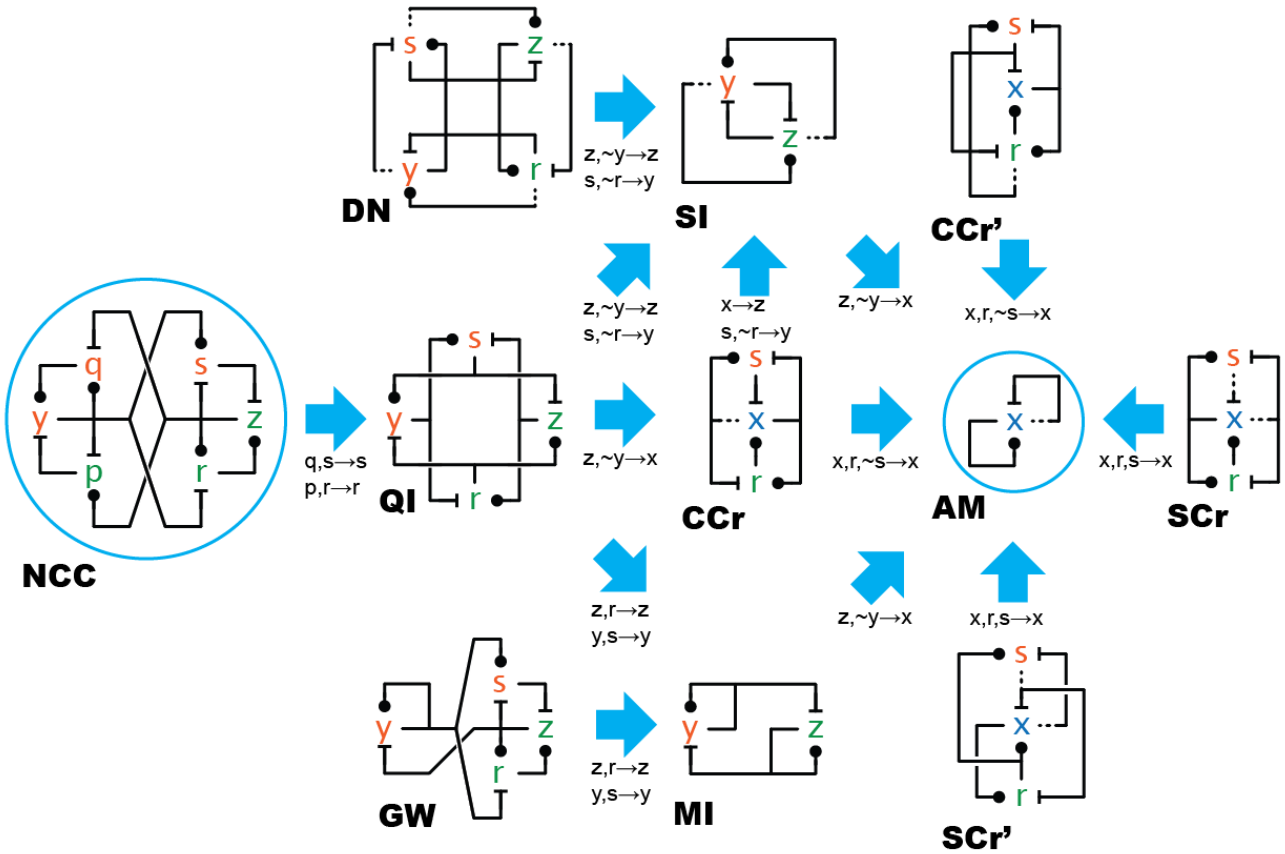
$$\mathbf{pr}[X_4, s3, \emptyset] = 0 \quad \mathbf{pr}[X_4, s3, X_3] = 3$$
$$\mathbf{pr}[X_5, s3, \emptyset] = 0 \quad \mathbf{pr}[X_5, s3, X_3] = 3$$

$$X_1 \xrightarrow{1} X_2$$
$$X_2 \xrightarrow{1} X_1$$
$$X_4 \xrightarrow{1} 2X_1 + X_3$$
$$X_5 \xrightarrow{1} 2X_2 + X_3$$
$$X_1 + X_3 \xrightarrow{3} X_4$$
$$X_2 + X_3 \xrightarrow{3} X_5$$
$$X_4 + X_3 \xrightarrow{3} X_5$$
$$X_5 + X_3 \xrightarrow{3} X_4$$

Based on [Derisavi et al., 2003]
and [Valmari & Franceschinis, 2010]

Case study: adaptation as evolution

"Zoo" of influence networks



From [Cardelli'14]

Influence networks: semantics

- Each node x corresponds to a triplet of chemical species x_0, x_1, x_2

high interm. low

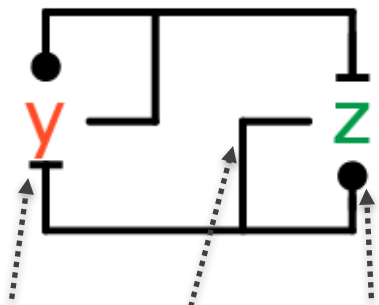
- Each triplet gives rise to four chemical reactions

$$x_0 + i \xrightarrow{k_1} i + x_1$$

$$x_1 + i \xrightarrow{k_2} i + x_2$$

$$x_2 + a \xrightarrow{k_3} a + x_1$$

$$x_1 + a \xrightarrow{k_4} a + x_0$$



inhibition activation

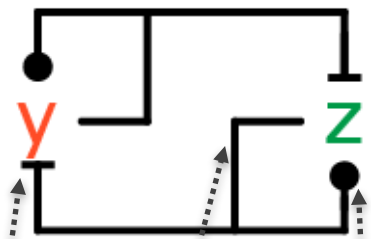
high output
(dashed otherwise)

$$y_0 + z_0 \to z_0 + y_1$$
$$y_1 + z_0 \to z_0 + y_2$$
$$y_2 + y_0 \to y_0 + y_1$$
$$y_1 + y_0 \to y_0 + y_0$$
$$z_2 + z_0 \to z_0 + z_1$$
$$z_1 + z_0 \to z_0 + z_0$$
$$z_0 + y_0 \to y_0 + z_1$$
$$z_1 + y_0 \to y_0 + z_2$$

Influence networks: semantics

- Each node x corresponds to a triplet of chemical species x_0, x_1, x_2

high interm. low



inhibition activation

high output (dashed otherwise)

- Each triplet gives rise to four chemical reactions

$$x_0 + i \xrightarrow{k_1} i + x_1$$

$$x_1 + i \xrightarrow{k_2} i + x_2$$

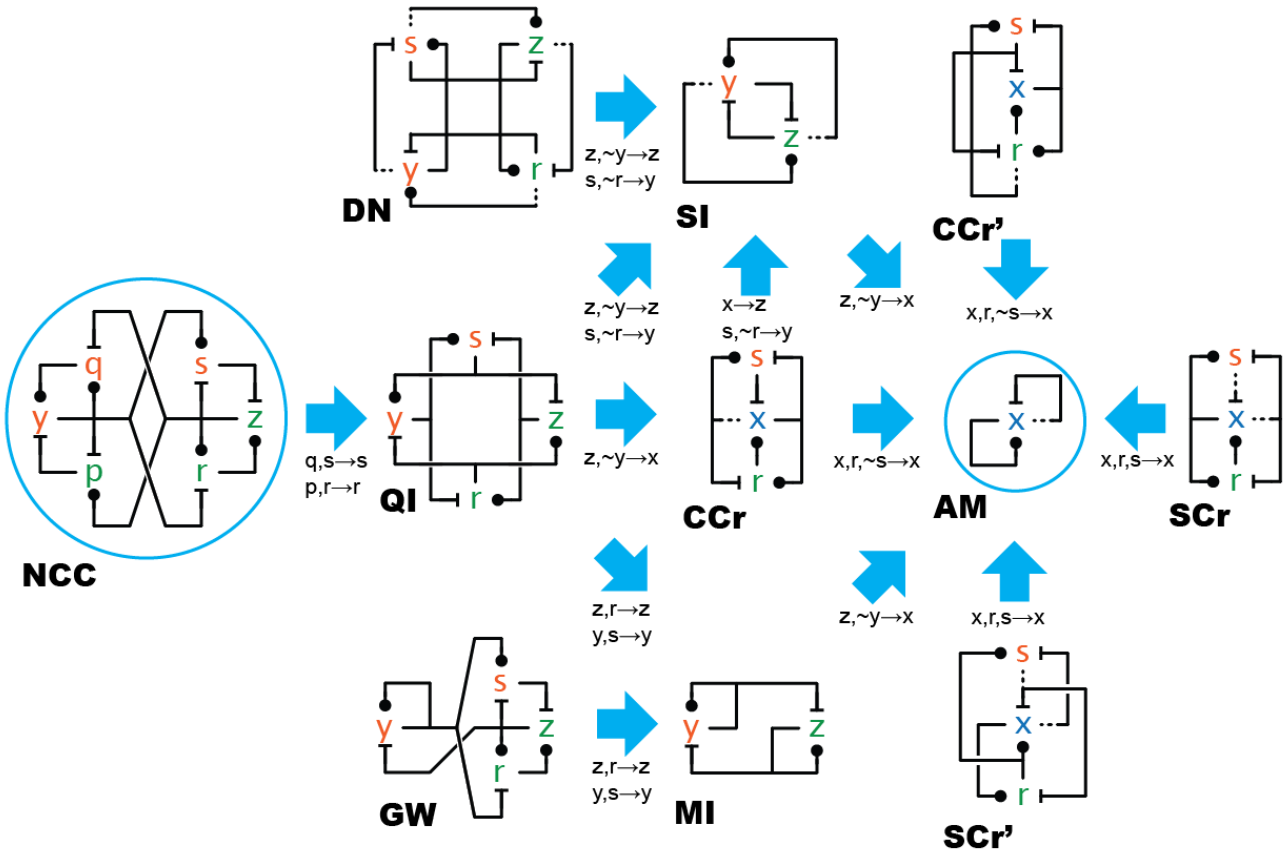
$$x_2 + a \xrightarrow{k_3} a + x_1$$

$$x_1 + a \xrightarrow{k_4} a + x_0$$

$$\frac{dX_0(t)}{dt} = -k_1 X_0(t) I(t) + \\ + k_4 X_1(t) A(t) + \\ + \dots$$

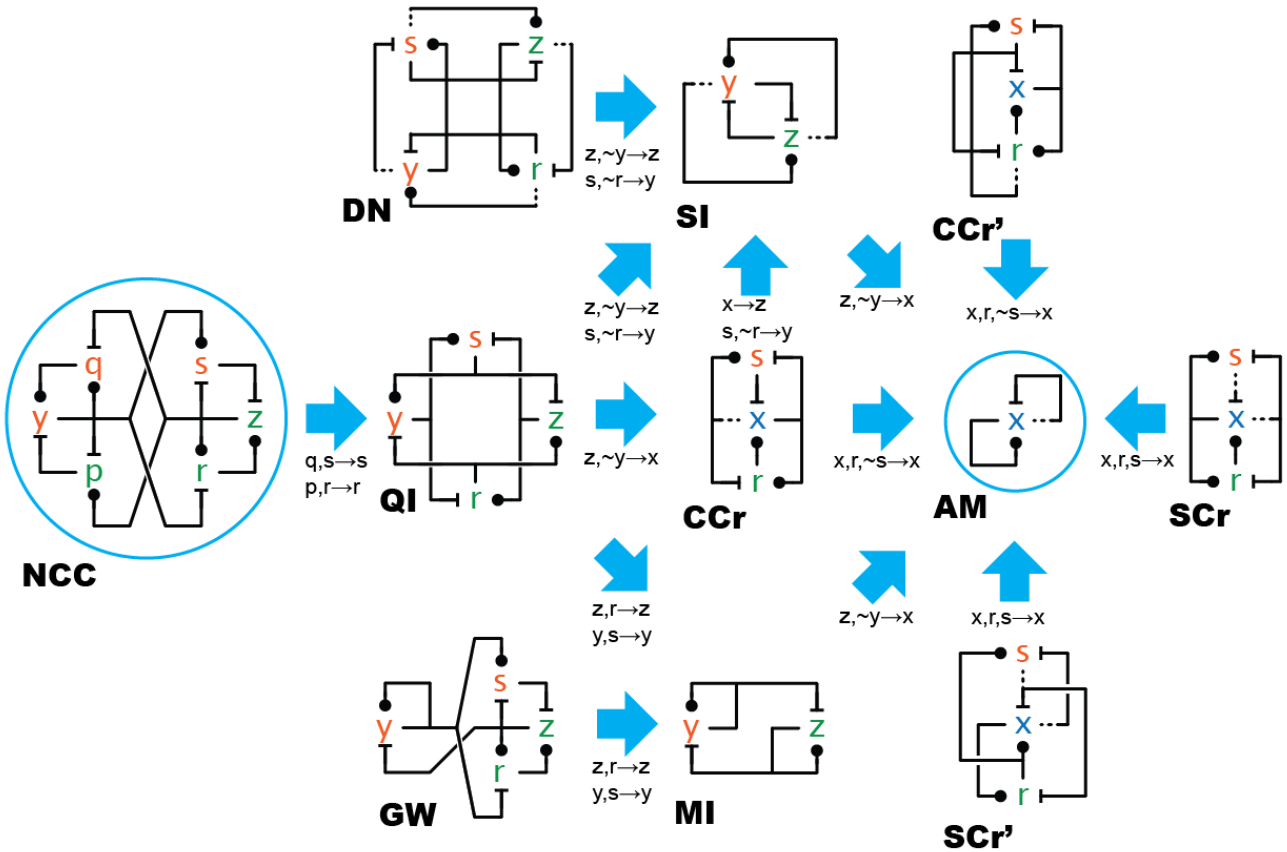
mass-action semantics (degree-two polynomials)

Emulation



- Emulation is a mapping between species of two influence networks such that related species have equal solutions at all time points

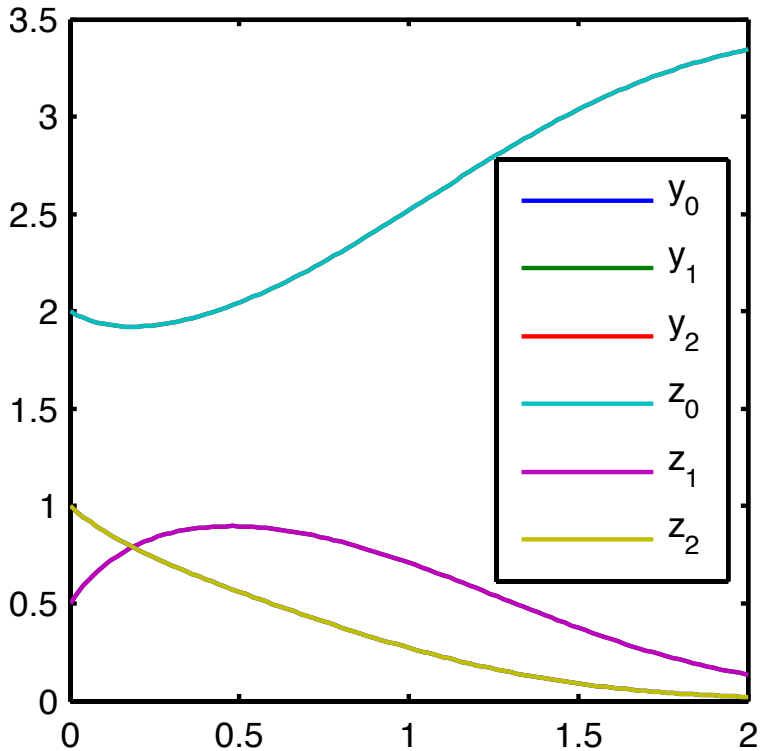
Emulation

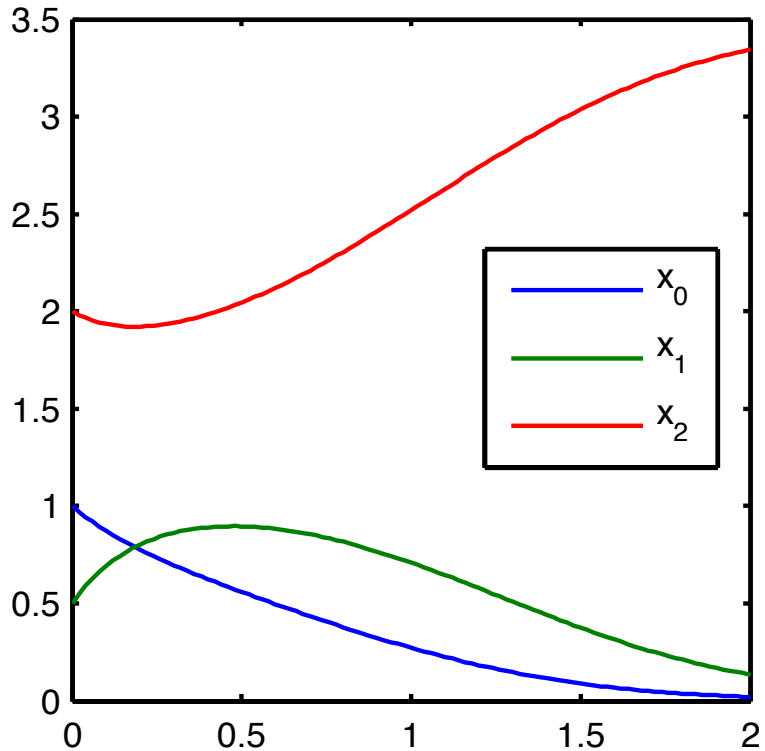


- For example $z, \sim y \to x$ for MI/AM means

$$z_0, y_2 \to x_0,\ z_2, y_0 \to x_2,\ z_1, y_1 \to x_1$$

Emulation

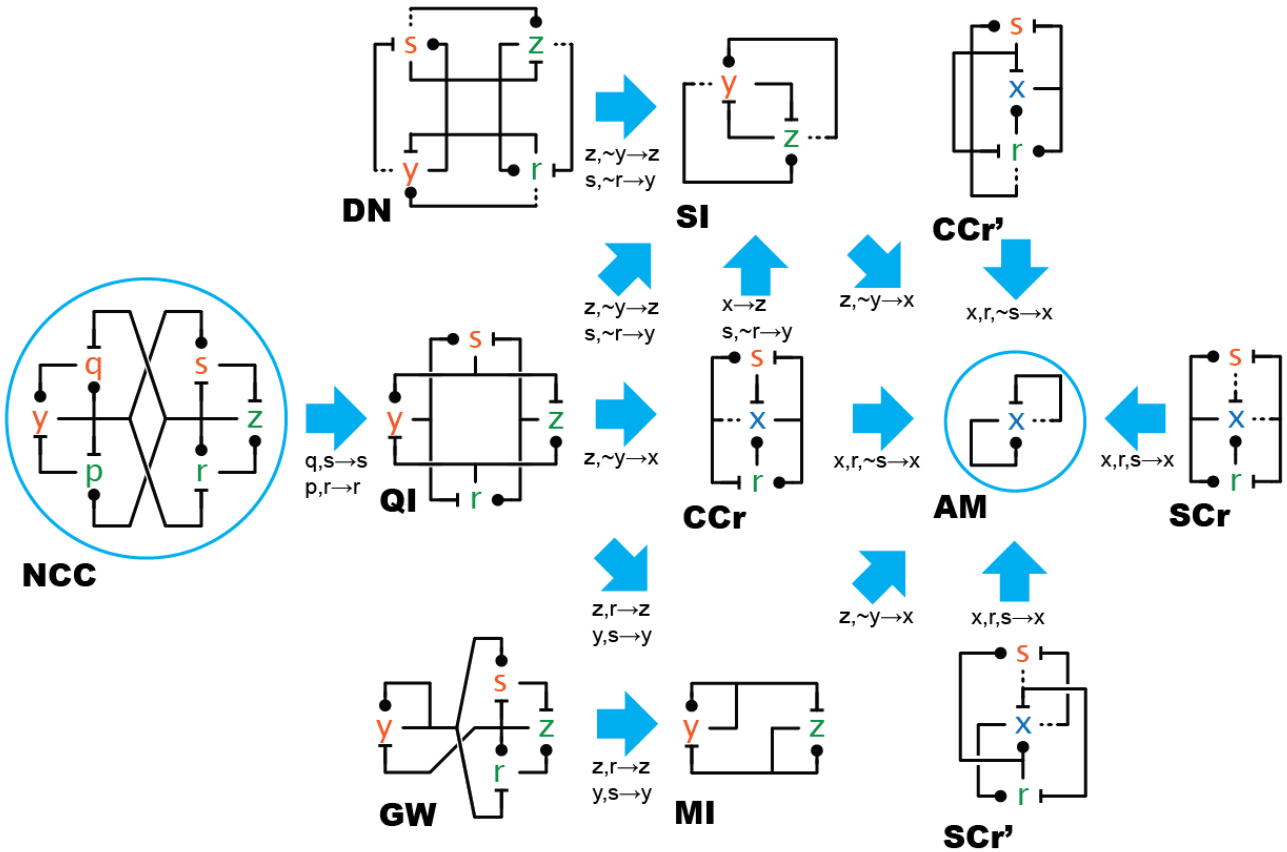




- For example $z, \sim y \rightarrow x$ for MI/AM means

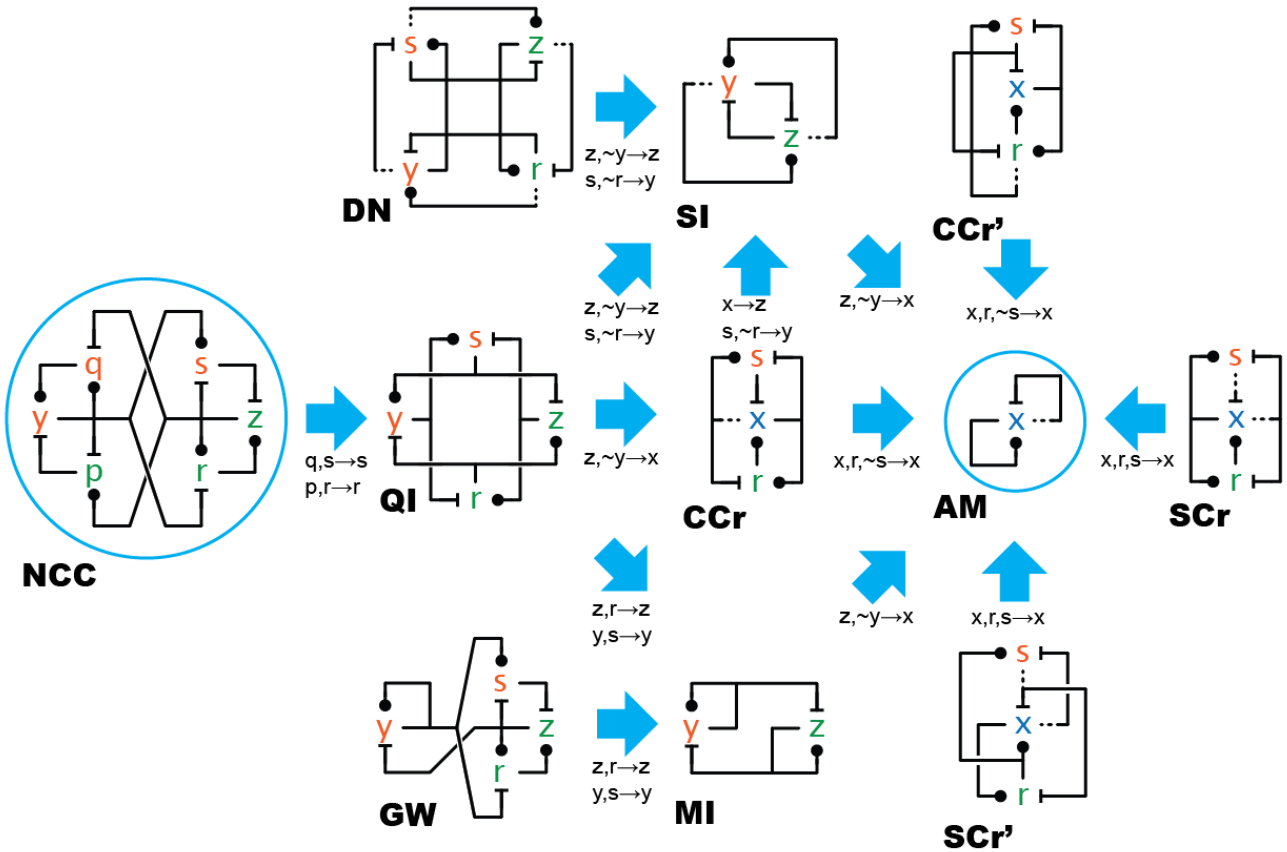
$$z_0, y_2 \rightarrow x_0,\ z_2, y_0 \rightarrow x_2,\ z_1, y_1 \rightarrow x_1$$

Emulation



- This can be discovered automatically by computing the largest backward bisimulation on the **union chemical reaction network**

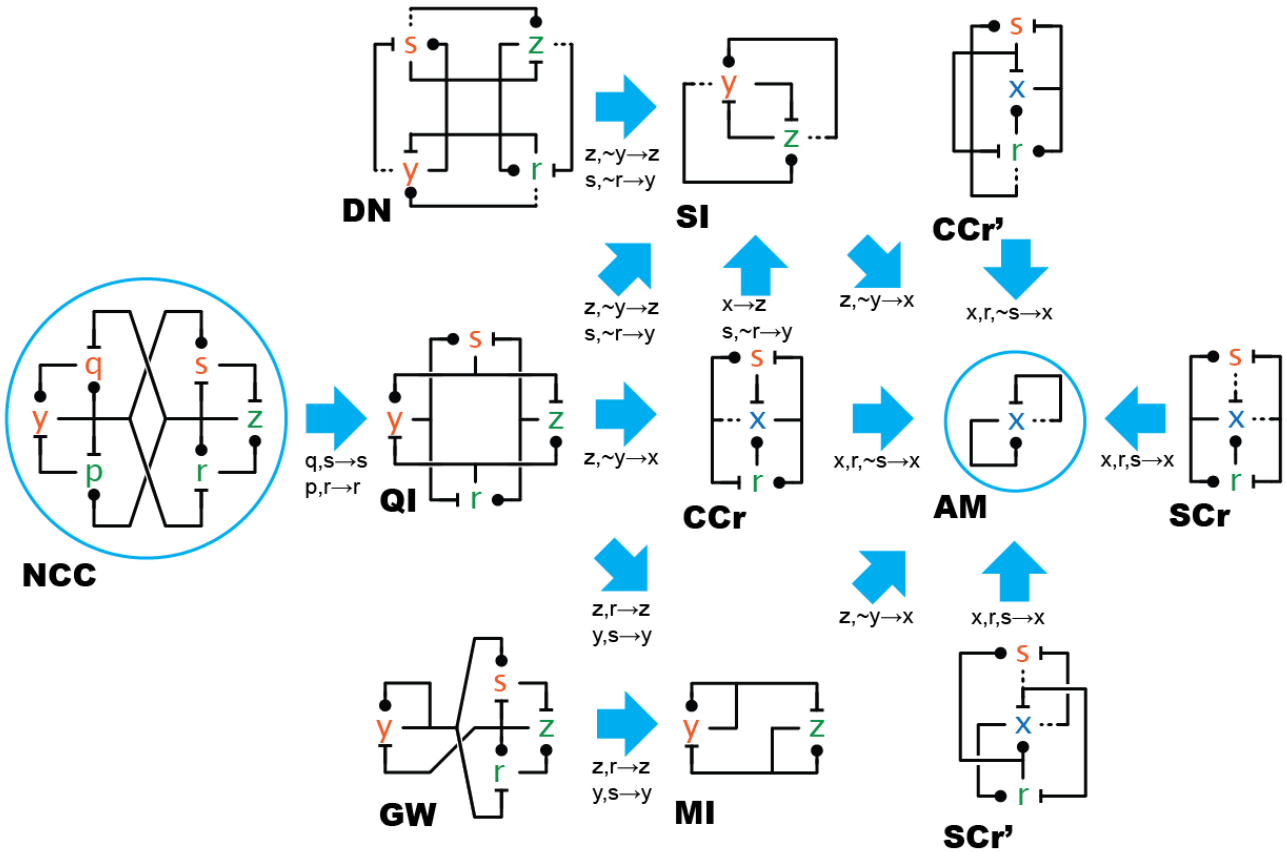
Emulation



- Largest backward bisimulation:

$$\{\{z_0, y_2, x_0\}, \{z_2, y_0, x_2\}, \{z_1, y_1, x_1\}\}$$

Analysis with differential equivalences



- Backward bisimulations found with mass-action assumptions carry over to other kinetic assumptions (Hill kinetics, checked with backward differential equivalence)

[POPL'16]

Numerical results

Original Model			Forward bisimulation		Forward equivalence	
ID	***Reactions***	***Vars***	***Vars***	***Time***	***Vars***	***Time***
M1	8620	745	745	0.65 s	105	> 2h
M2	3680	354	354	0.28 s	105	~ 1 h
M3	4944	411	411	0.13 s	47	10 min
M4	3477	348	348	0.25 s	215	~1.5 h

[POPL'16]

Numerical results

Original Model			Forward bisimulation		Forward equivalence	
ID	*Reactions*	*Vars*	*Vars*	*Time*	*Vars*	*Time*
M1	8620	**745**	**745**	0.65 s	**105**	> 2h
M2	3680	354	354	0.28 s	105	~ 1 h
M3	4944	411	411	0.13 s	47	10 min
M4	3477	348	348	0.25 s	215	~1.5 h

[POPL'16]

- Forward bisimulation may miss reductions

Numerical results

Original Model			Forward bisimulation		Forward equivalence	
ID	*Reactions*	*Vars*	*Vars*	*Time*	*Vars*	*Time*
M1	8620	745	745	**0.65 s**	105	**> 2h**
M2	3680	354	354	0.28 s	105	~ 1 h
M3	4944	411	411	0.13 s	47	10 min
M4	3477	348	348	0.25 s	215	~1.5 h

[POPL'16]

- Forward bisimulation may miss reductions
- But forward equivalence is significantly more time consuming

Numerical results

Original Model			Forward bisimulation		Backward bisimulation	
ID	***Reactions***	***Vars***	***Vars***	***Time***	***Vars***	***Time***
CRN1	3,538,944	262,146	222	7.49 s	222	12 s
CRN5	194,054	14,531	10,855	0.40 s	6,634	0.6 s
CRN13	24	18	18	4 ms	7	4 ms
AFF2	8,814,880	**1,270,433**	**160,951**	~ 10 min	639,509	~ 3 min

[TACAS'16]

Numerical results

Original Model			Forward bisimulation		Backward bisimulation	
ID	***Reactions***	***Vars***	***Vars***	***Time***	***Vars***	***Time***
CRN1	**3,538,944**	**262,146**	222	**7.49 s**	222	**12 s**
CRN5	194,054	14,531	10,855	0.40 s	6,634	0.6 s
CRN13	24	18	18	4 ms	7	4 ms
AFF2	8,814,880	1,270,433	160,951	~ 10 min	639,509	~ 3 min

[TACAS'16]

- Bisimulation algorithms scale well
  (original CRN1 could not be solved on our machines)

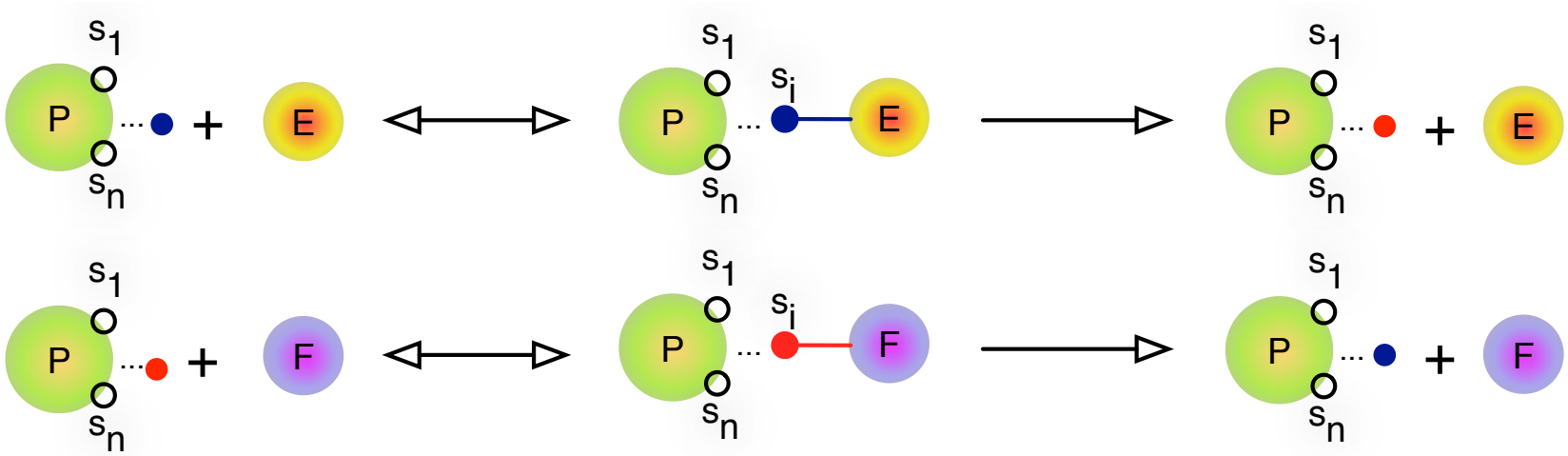
Numerical results

Original Model			Forward bisimulation		Backward bisimulation	
ID	***Reactions***	***Vars***	***Vars***	***Time***	***Vars***	***Time***
CRN1	3,538,944	262,146	222	7.49 s	222	12 s
CRN5	194,054	14,531	**10,855**	0.40 s	**6,634**	0.6 s
CRN13	24	18	18	4 ms	7	4 ms
AFF2	8,814,880	1,270,433	160,951	~ 10 min	639,509	~ 3 min

[TACAS'16]

- Bisimulation algorithms scale well
  (original CRN1 could not be solved on our machines)
- Forward and backward bisimulation are not comparable

Physical interpretations



$$P(b,b,\ldots,b) + E \xrightarrow{k_1} P(E,b,\ldots,b) \qquad P(r,b,\ldots,b) + F \xrightarrow{k_2} P(F,b,\ldots,b)$$

$$P(b,b,\ldots,b) + E \xrightarrow{k_1} P(b,E,\ldots,b) \qquad P(b,r,\ldots,b) + F \xrightarrow{k_2} P(b,F,\ldots,b)$$

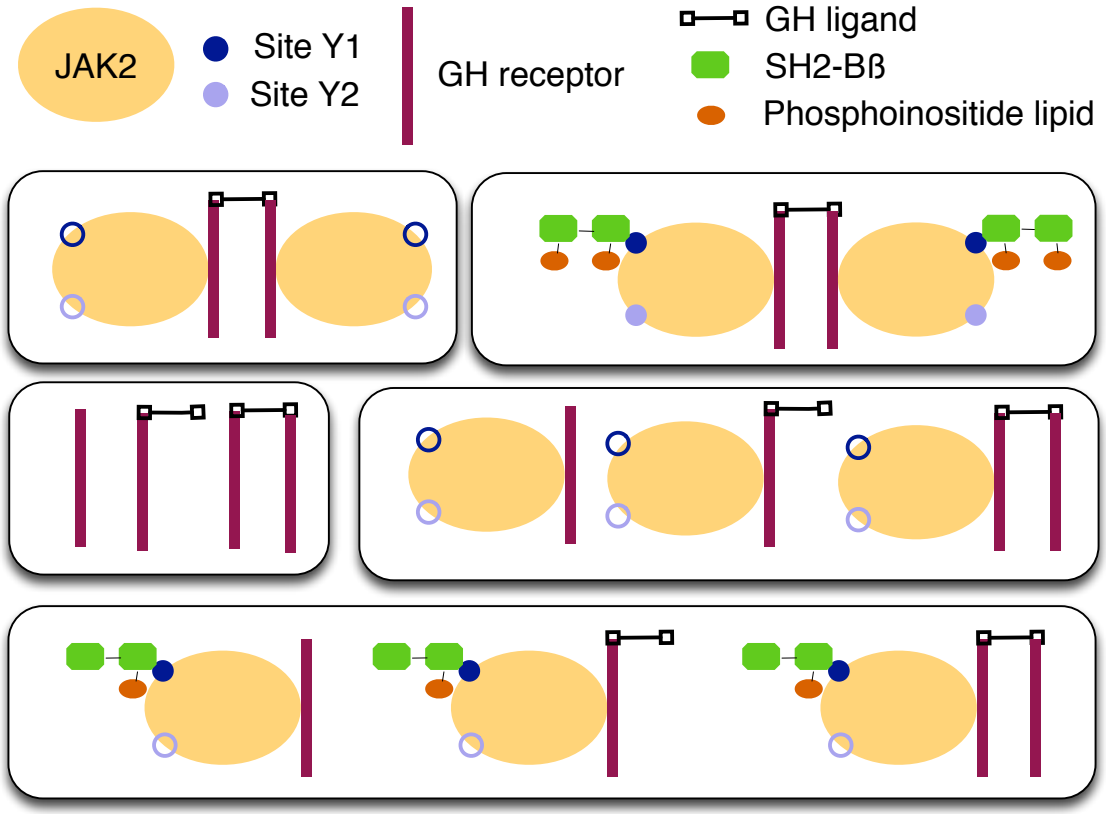
$$\ldots \qquad\qquad \ldots$$

$$P(b,b,\ldots,b) + E \xrightarrow{k_1} P(b,b,\ldots,E) \qquad P(b,b,\ldots,r) + F \xrightarrow{k_2} P(b,b,\ldots,F)$$

$$\ldots \qquad\qquad \ldots$$

- Symmetry reduction in protein interaction networks

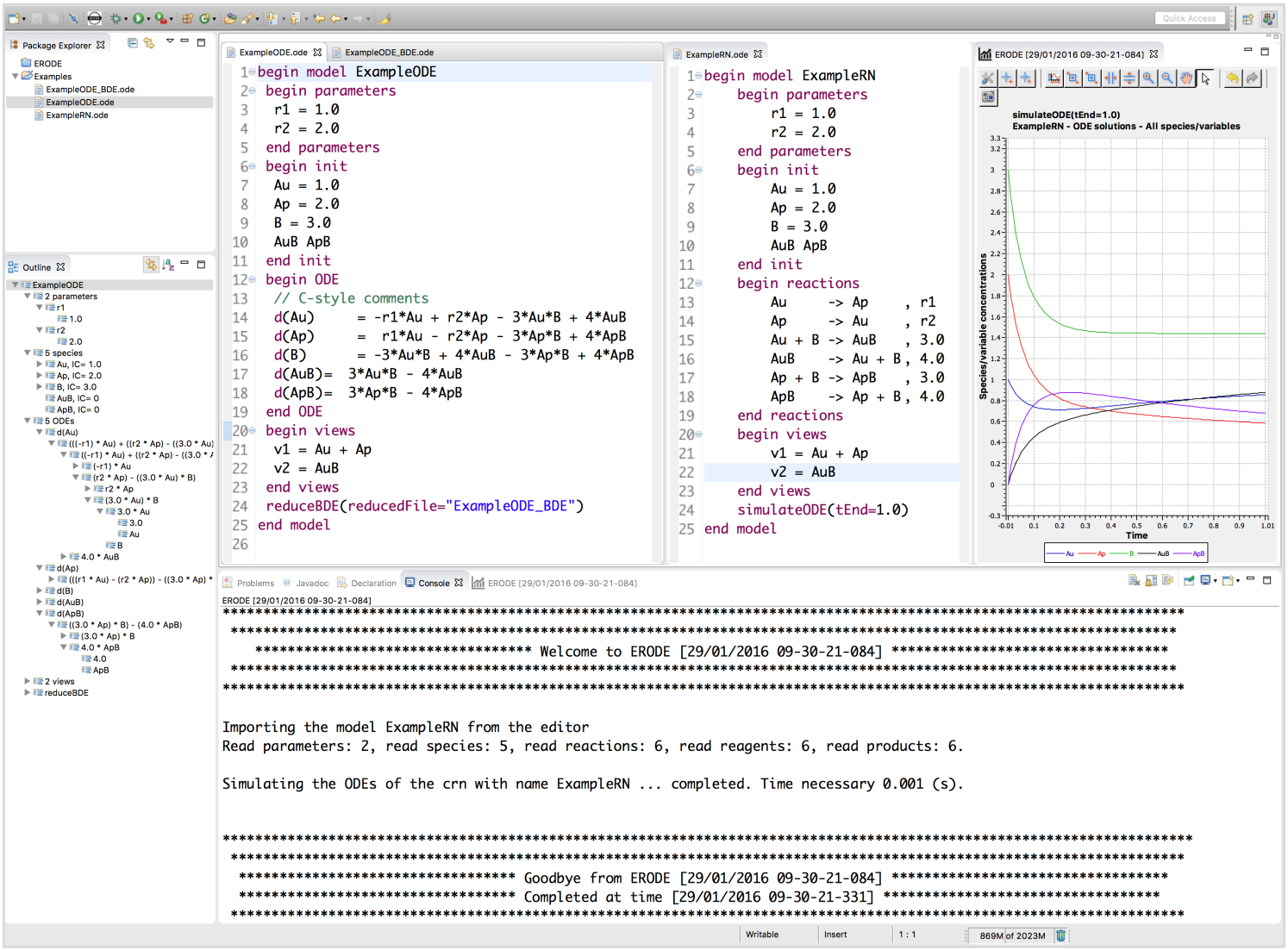
Physical interpretations



[Barua et al. '09]

- Endocytosed complexes are equivalent when they have the same structure up to the conformation of the GH ligand

Tool support: ERODE



Discussion

- Equivalences allow to automatically reduce large systems of nonlinear ODE **exactly**
  - Encoding into SMT for general case: complete but slower (although it can be made more efficient)
  - Efficient partition refinement for multivariate polynomials of degree at most two:
    - Quite a large class on its own (contains affine systems and mass-action CRNs)
    - Sometimes nonlinear systems are handled by first expanding into such a class of polynomials

Discussion

- Still insisting on **exact reductions**:
  - Extend bisimulation results to arbitrary-degree polynomials
  - Go beyond partition refinement? It is not possible to encode arbitrary constraints on the initial partition to be refined [LICS'16]
  - For the forward case, go beyond sum-of-variables? Arbitrary linear transformations may give further compressions

Discussion

- **Approximate variants** are highly desired
  - Problem is how to compute error bounds a priori, without solving the original model first
  - Some techniques for nonlinear systems have been developed **but**
    1. No algorithm is available to compute candidate partition [DSN'13, TAC'16, Bortolussi & Gast'16]
    2. Bounds tend to be loose because only part of the information about the structure of the system is exploited [DSN'13, TAC'16]

Some related work

- Fragmentation/lumpability in Kappa [Danos/Feret]: domain specific
- Exact and ordinary lumpability for Markov chains [Buchholz]: special case of ODE equivalences
- Lumping algorithms for Markov chains [Derisavi et al., Valmari & Franceschinis]: special cases of minimization algorithms for forward and backward RN bisimulation
- Exact lumpability in CRNs [Li & Rabitz]: no algorithm available; forward bisimulation special case
- Exact bisimulations for control systems [Pappas]: preservation of the controllability subspace
- Model order reduction (MOR) techniques [Antoulas]: mostly for linear systems, approximate without error bounds

References (1/2)

- G. Li, and H. Rabitz: *A general analysis of exact lumping in chemical kinetics*. **Chemical Engineering Science** 44 (1989) 1413–1430
- P. Buccholz: *Exact and Ordinary Lumpability in Finite Markov Chains.* **Journal of Applied Probability,** 31 59–75 (1994)
- S. Derisavi,, H. Hermanns, W. Sanders: *Optimal state-space lumping in Markov chains*. **Inf. Process. Lett.** 87 (2003) 309–315
- G. J. Pappas: *Bisimilar linear systems*. **Automatica**, 39(12):2035–2047, 2003.
- A. Antoulas: *Approximation of Large-Scale Dynamical Systems*. Advances in Design and Control. SIAM, 2005.
- J. Hillston: *Fluid Flow Approximation of PEPA Models.* **QEST 2005**
- D. Barua, J. R. Faeder, and J. M. Haugh: *A bipolar clamp mechanism for activation of jak-family protein tyrosine kinases*. **PLoS Computational Biology**, 5(4), 2009
- V. Danos, J. Feret, W. Fontana, R. Harmer, and J. Krivine: *Abstracting the differential semantics of rule-based models: Exact and automated model reduction*. **LICS 2010**

References (2/2)

- A.Valmari, G Franceschinis: Simple *O(mlogn)* time Markov Chain lumping. **TACAS 2010**
- G. Iacobelli, M. Tribastone: *Lumpability of Fluid Models with Heterogeneous Agent Types*. **DSN 2013**
- L. Cardelli: *Morhpisms of reaction networks that couple structure to function.* **BMC Systems Biology**, 2014
- L. Cardelli, M. Tribastone, M. Tschaikowski, A. Vandin: *Forward and backward bisimulations for chemical reaction networks*. **CONCUR 2015**
- L. Cardelli, M. Tribastone, M. Tschaikowski, A. Vandin: *Symbolic computation of differential equivalences*, **POPL 2016**
- L. Cardelli, M. Tribastone, M. Tschaikowski, A. Vandin. *Efficient syntax-driven lumping of differential equations*, **TACAS 2016**
- L. Cardelli, M. Tribastone, M. Tschaikowski, A. Vandin. *Comparing Chemical Reaction Networks: A Categorical and Algorithmic Perspective*, **LICS 2016**
- M. Tschaikowski, M. Tribastone. *Approximate reduction of heterogeneous models with differential hulls*, **Transactions on Automatic Control**,**2016**
- L. Bortolussi and N. Gast: *Mean Field Approximation of Uncertain Stochastic Models*. **DSN 2016**