Molecules as Automata

Representing Biochemical Systems as Collectives of Interacting Automata

Luca Cardelli

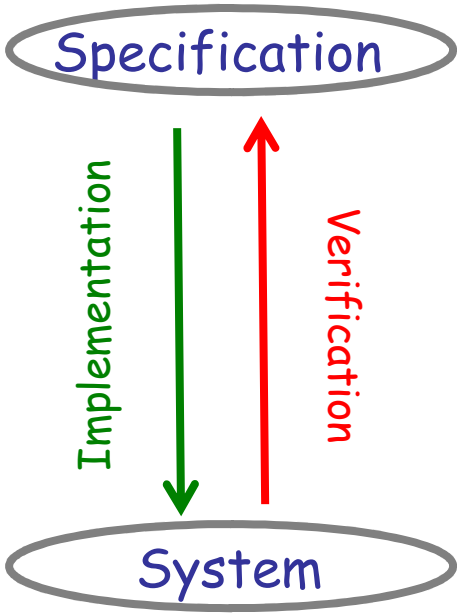
Microsoft Research

NEWSYNBIO, Paris, 2008-06-27

http://LucaCardelli.name

Scientific Method vs. Engineering Method

Direct Engineering

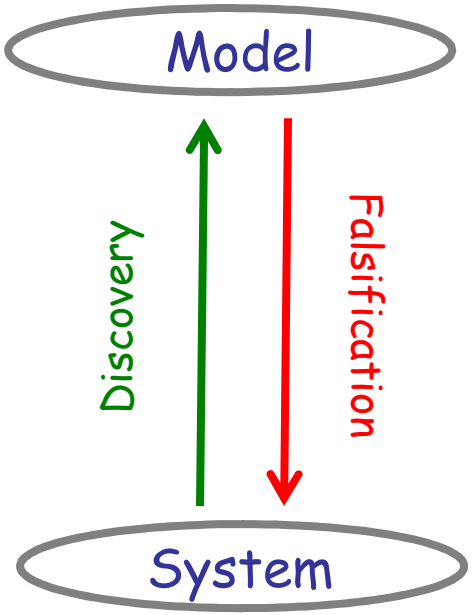


Engineering Method

<- Abstract ->

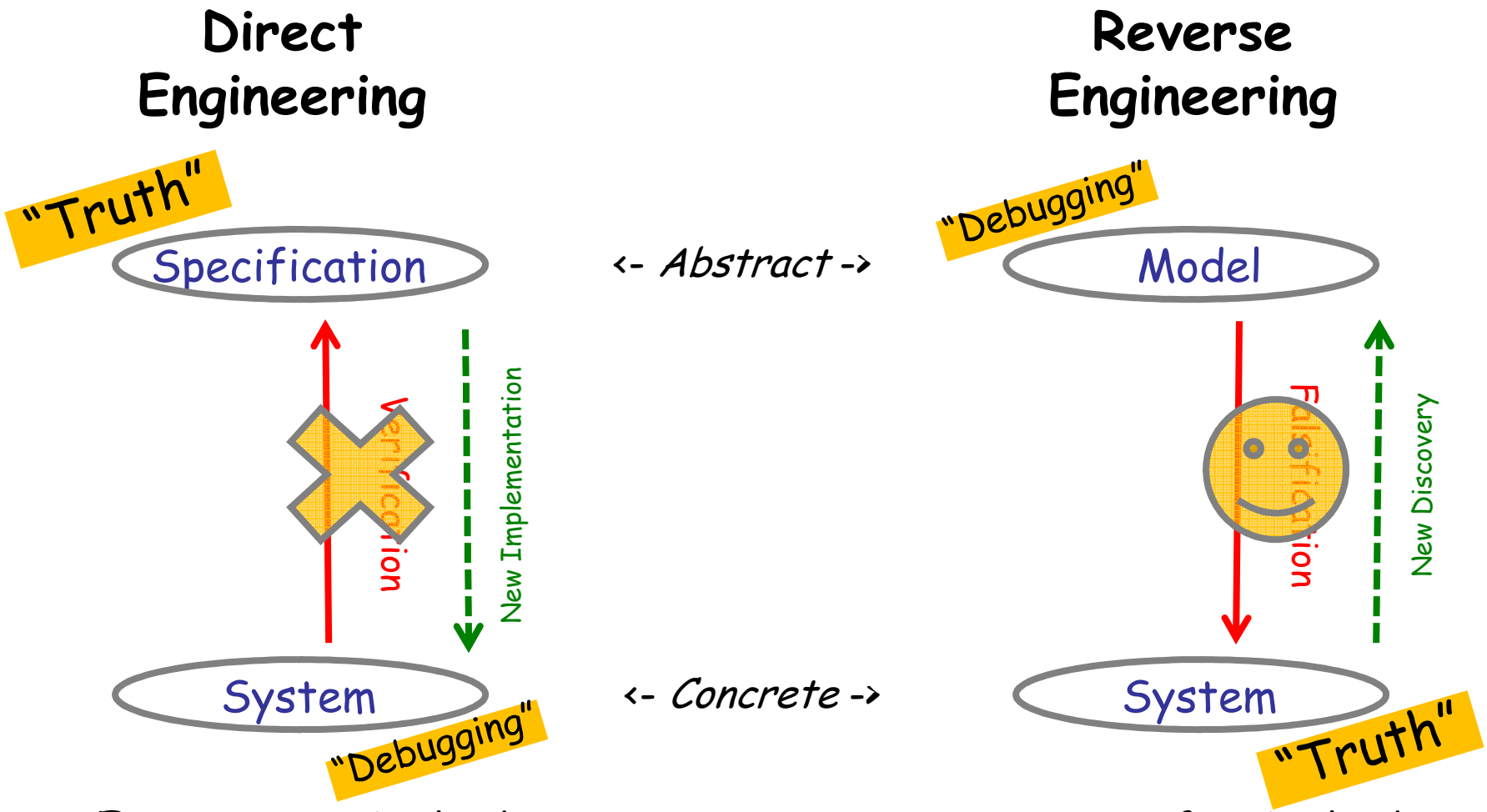
<- Concrete ->

Reverse Engineering

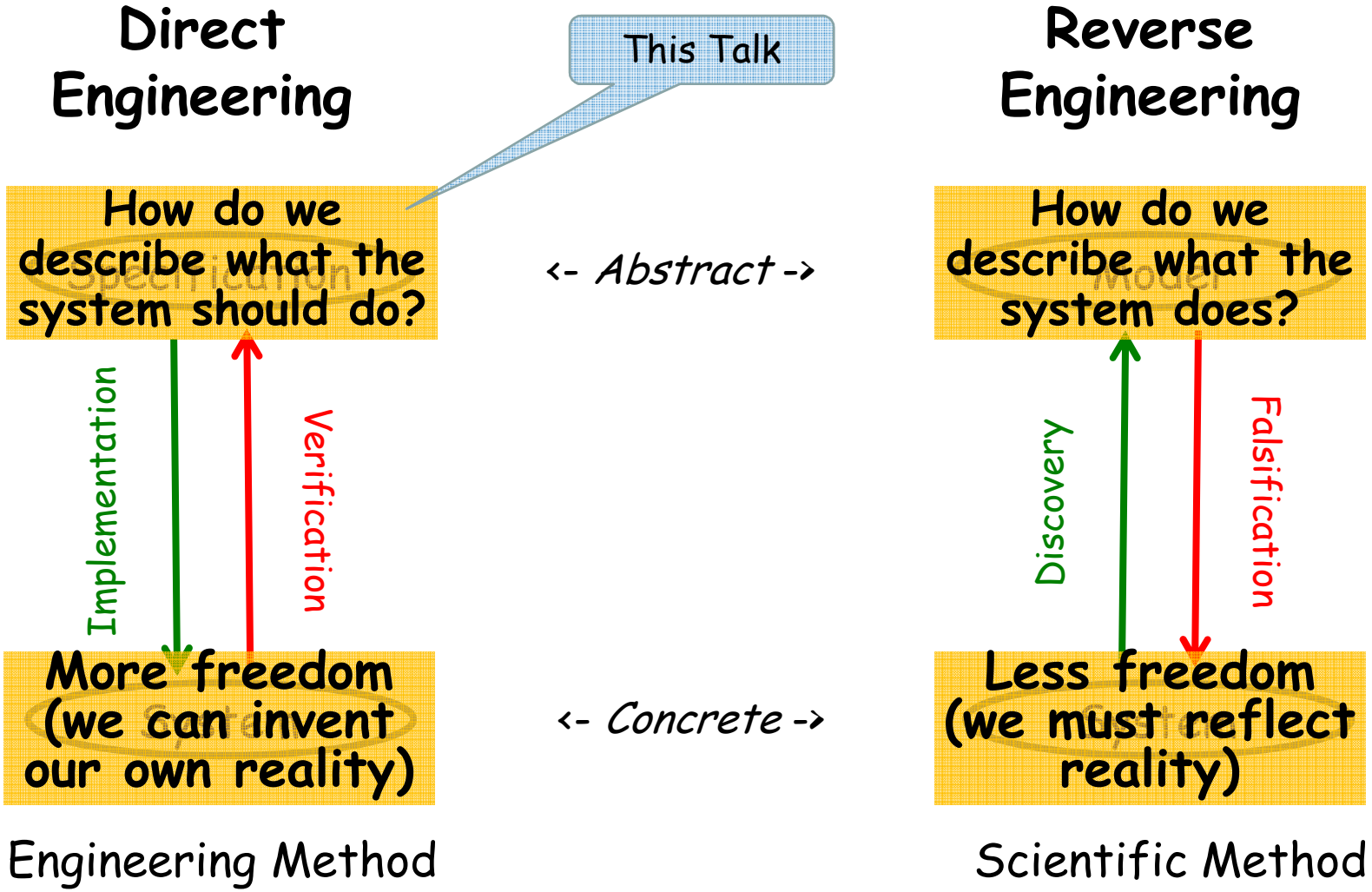


Scientific Method

Scientific Method vs. Engineering Method

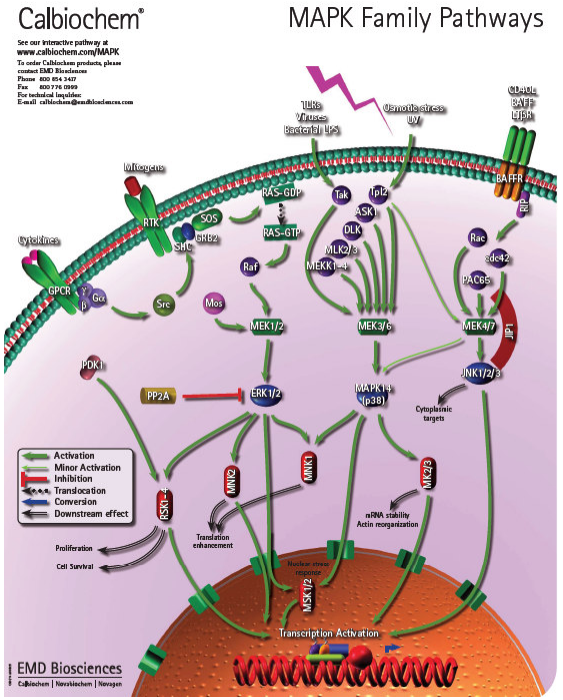


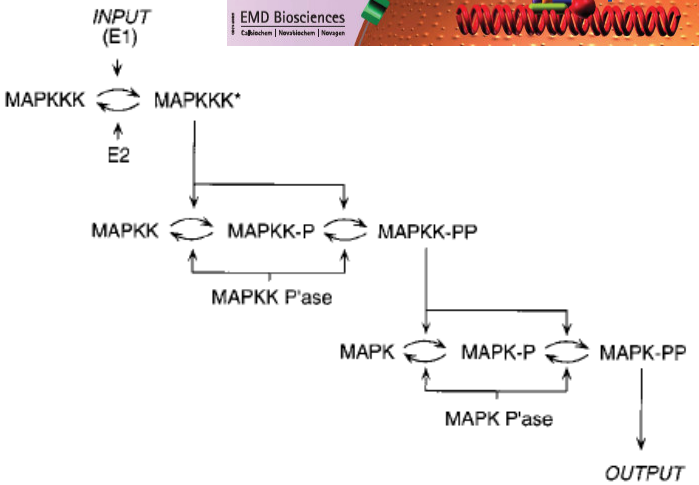
Scientific Method vs. Engineering Method



Motivation: Cells Compute

- No survival without computation!
  - Finding food
  - Avoiding predators
- *How* do they compute?
  - Unusual computational paradigms.
  - Proteins: do they work like electronic circuits?
  - Genes: what kind of software is that?
- Signaling networks
  - Clearly "information processing"
  - They are "just chemistry": molecule interactions
  - But what are their principles and algorithms?
- Complex, higher-order interactions
  - MAPKKK = MAP Kinase Kinase Kinase:
    that which operates on that which operates on that which operates on protein.
- General models of biological computation
  - What are the appropriate ones?





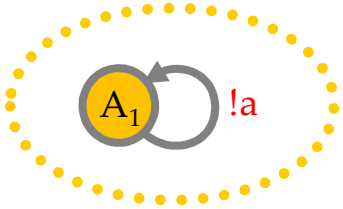
Ultrasensitivity in the mitogen-activated protein cascade, Chi-Ying F. Huang and James E. Ferrell, Jr., 1996, *Proc. Natl. Acad. Sci. USA*, 93, 10078-10083.

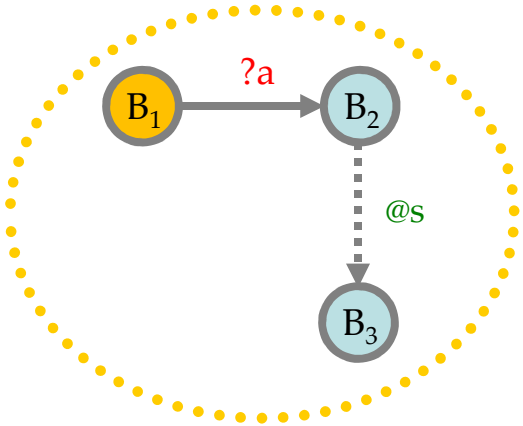
(Macro-) Molecules as (Interacting) Automata

- Concurrent (math is based on processes, not functions)
- Asynchronous (no global clock)
- Stochastic (or nondeterministic)
- Stateful (e.g. phosphorylation state)
- Discrete (transitions between states)
- Interacting (an "interaction" can be pretty much anything you want that changes molecular state)

- Based on work on process algebra and biological modeling; see references in related papers.

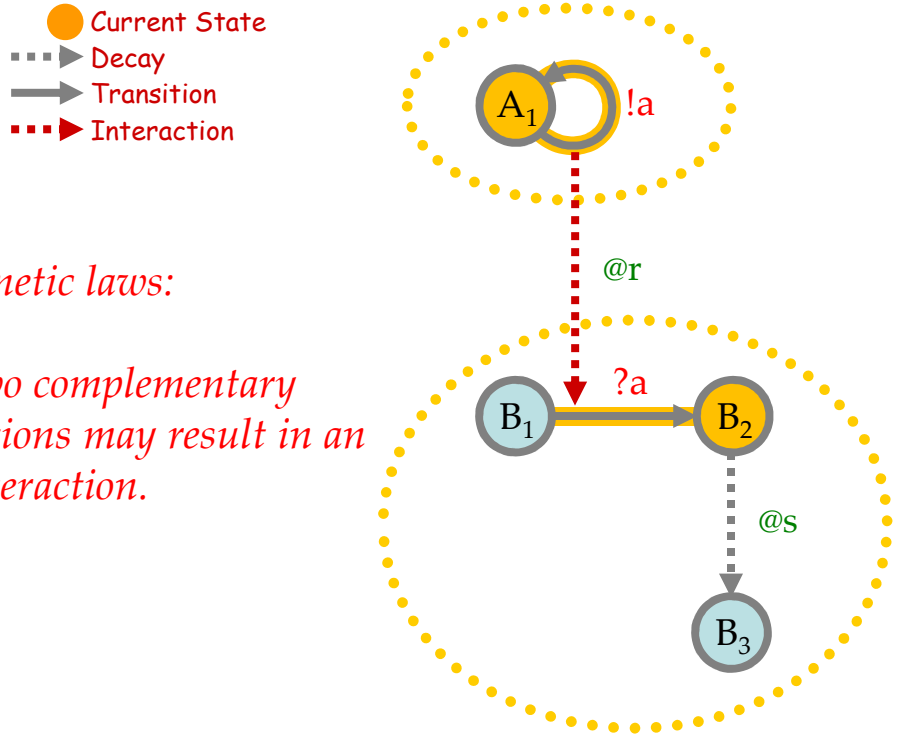
Interacting Automata





A_1	is a *state*
a	is a *channel* i.e. a named *interaction interface* (e.g. a surface patch)
?,!	indicate any *complementarity* of interaction (e.g. charge)
?a, !a	indicate *complementary actions*,
@r, @s	are rates

Interacting Automata

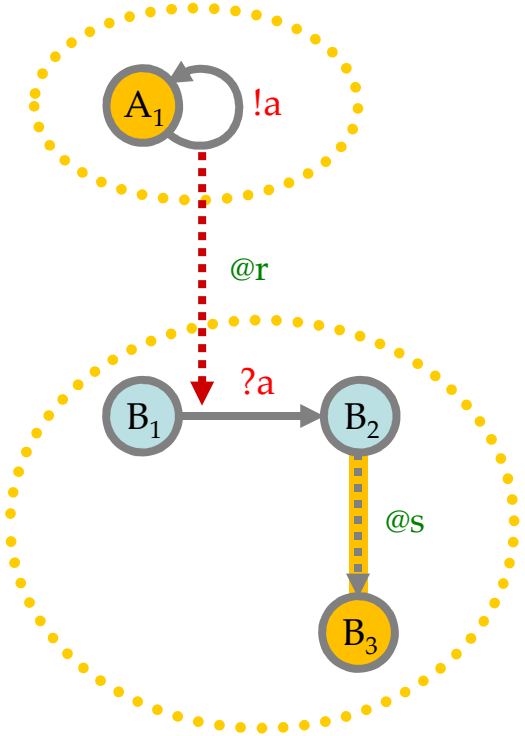


Kinetic laws:

Two complementary actions may result in an interaction.

A_1	is a *state*
a	is a *channel* i.e. a named *interaction interface* (e.g. a surface patch)
?,!	indicate any *complementarity* of interaction (e.g. charge)
?a, !a	indicate *complementary actions*, joined by an interaction arrow ⋯▶
@r, @s	are rates

Interacting Automata



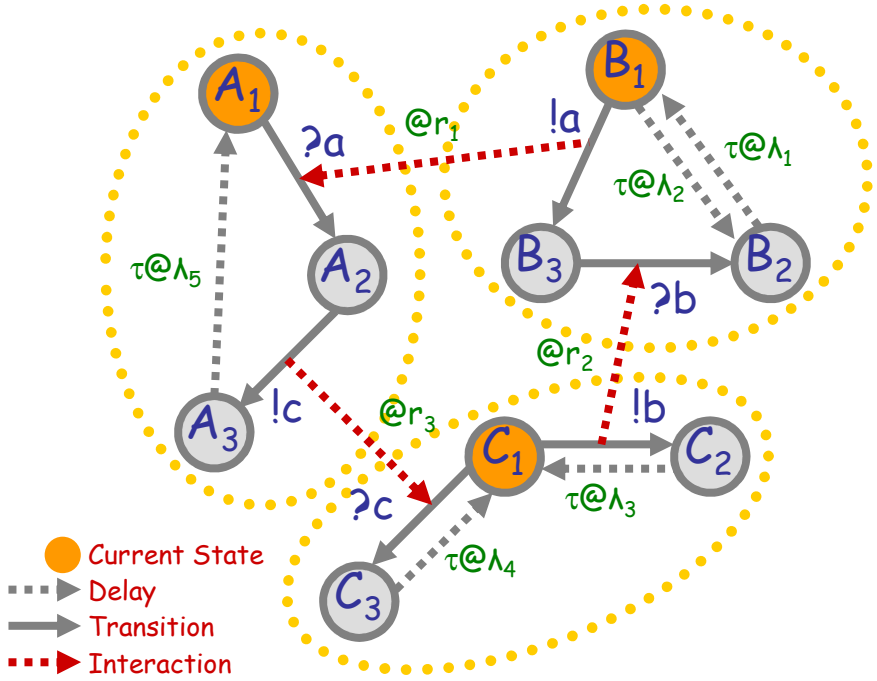
Kinetic laws:

Two complementary actions may result in an interaction.

A decay may happen spontaneously.

A_1	is a *state*
a	is a *channel* i.e. a named *interaction interface* (e.g. a surface patch)
?,!	indicate any *complementarity* of interaction (e.g. charge)
?a, !a	indicate *complementary actions*, joined by an interaction arrow ⋯▶
@r, @s	are rates

Interacting Automata



Interactions have rates. Actions DO NOT have rates.

The equivalent process algebra model

new a@r_1
new b@r_2
new c@r_3

Communication channels

$A_1 = ?a;\ A_2$
$A_2 = !c;\ A_3$
$A_3 = \tau@\lambda_5;\ A_1$

$B_1 = \tau@\lambda_2;\ B_2 + !a;\ B_3$
$B_2 = \tau@\lambda_1;\ B_1$
$B_3 = ?b;\ B_2$

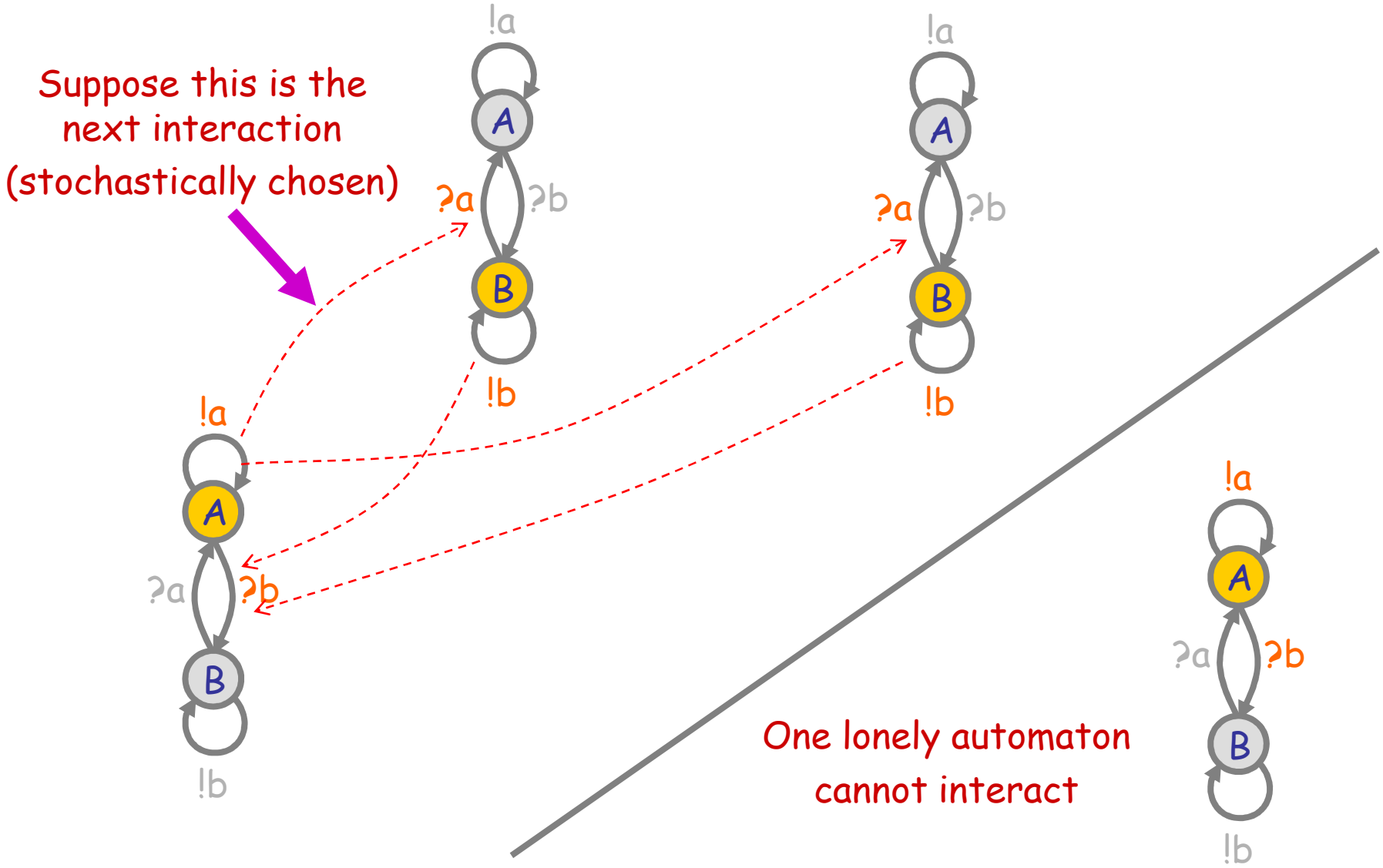
$C_1 = !b;\ C_2 + ?c;\ C_3$
$C_2 = \tau@\lambda_3;\ C_1$
$C_3 = \tau@\lambda_4;\ C_2$

Automata

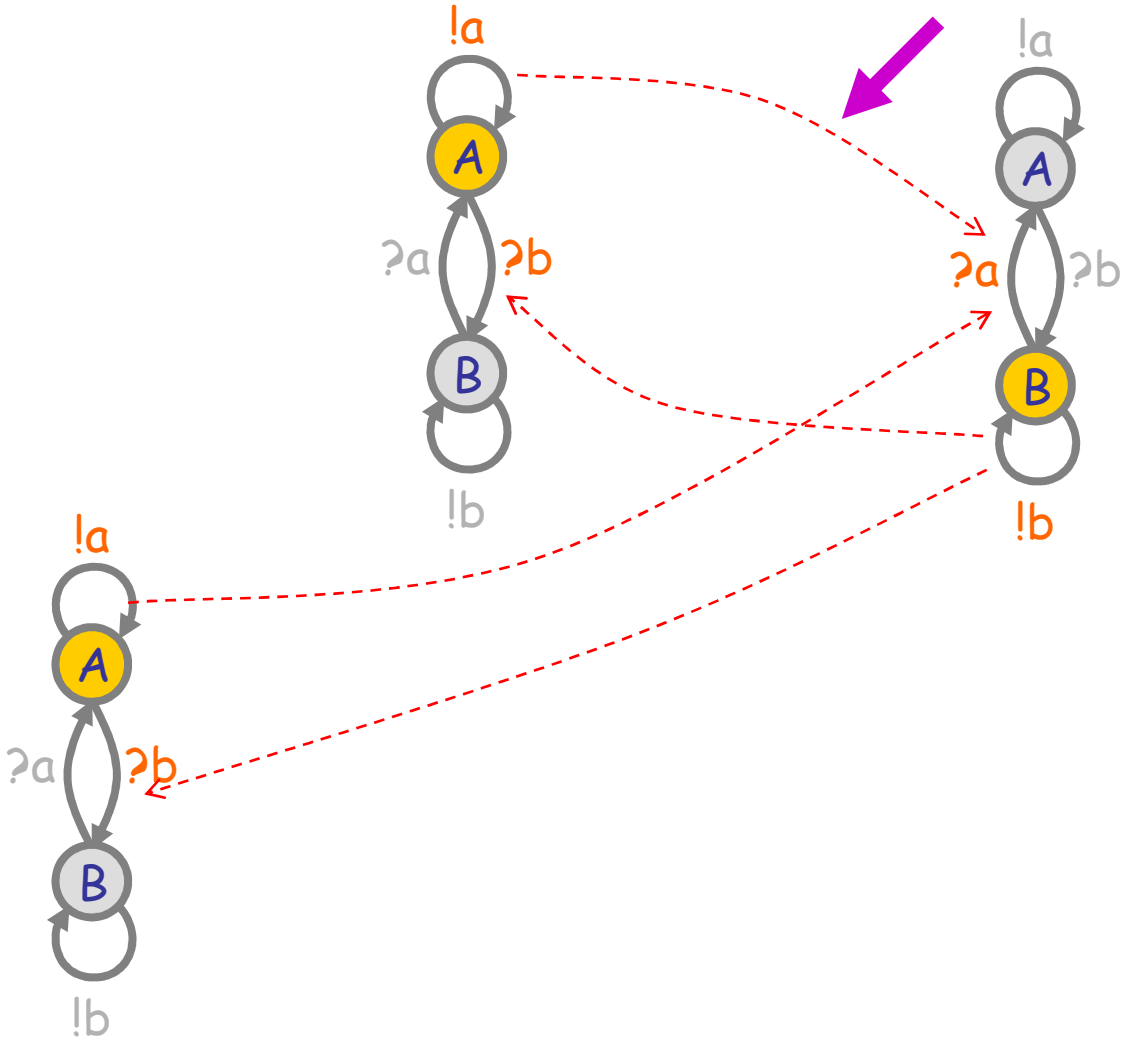
$A_1 \mid B_1 \mid C_1$

The system and initial state

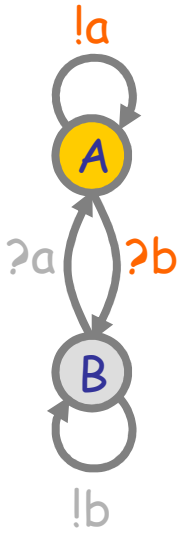
Interactions in a Population

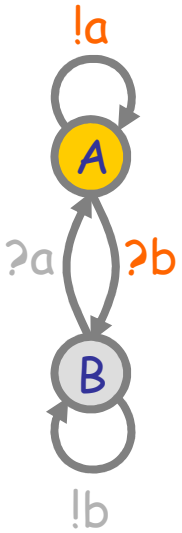


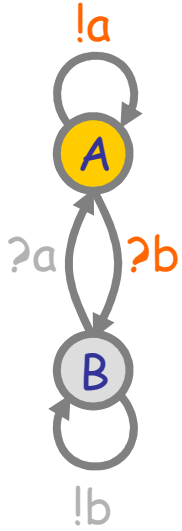
Interactions in a Population



Interactions in a Population

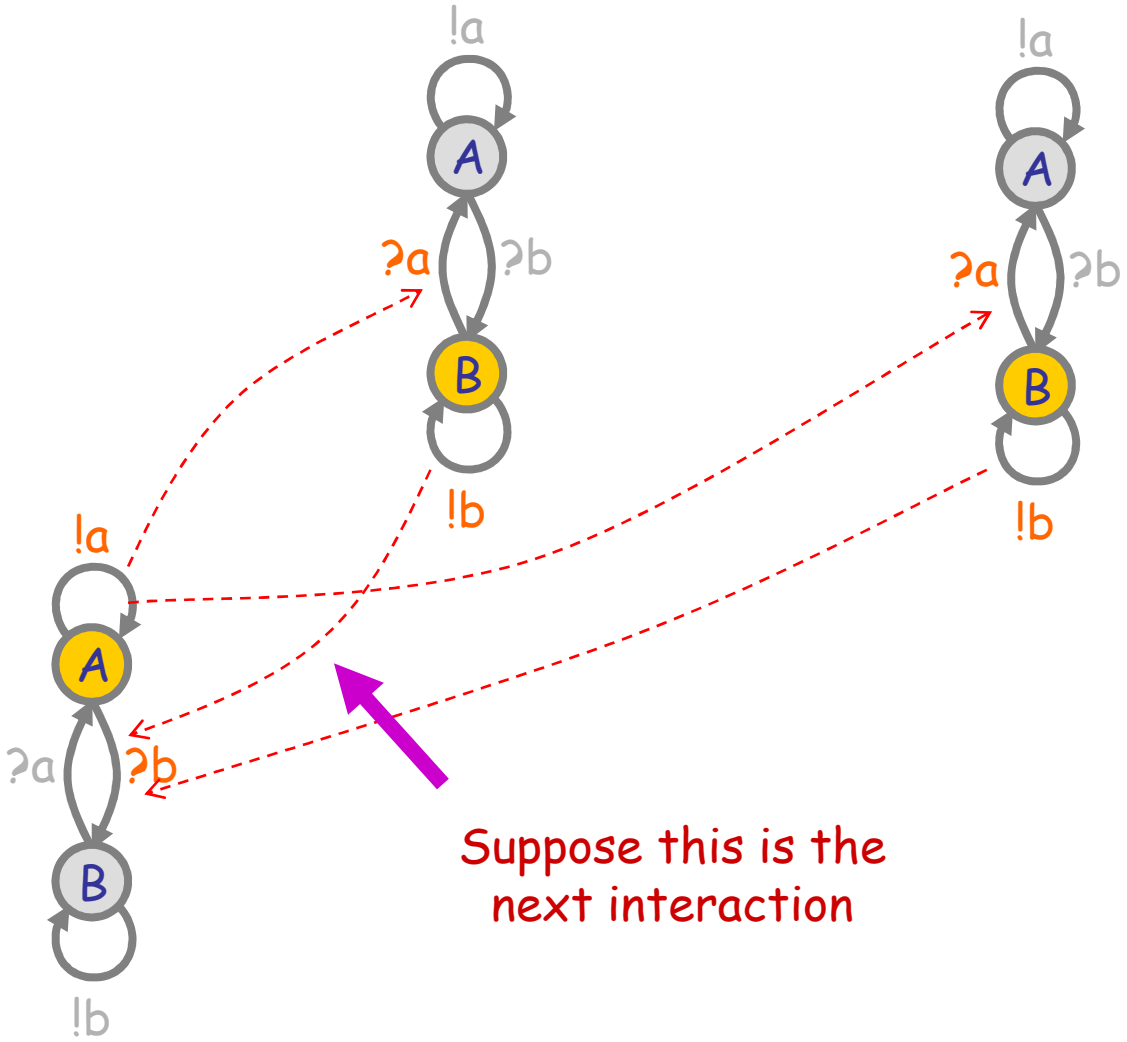




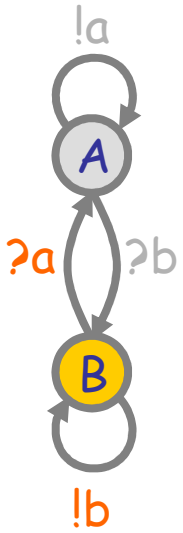


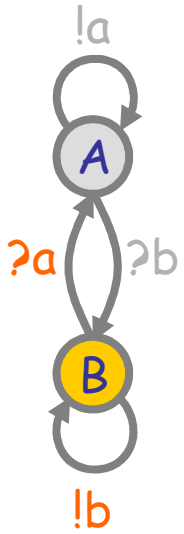
All-A stable population

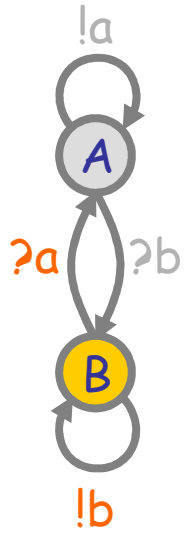
Interactions in a Population (2)



Interactions in a Population (2)



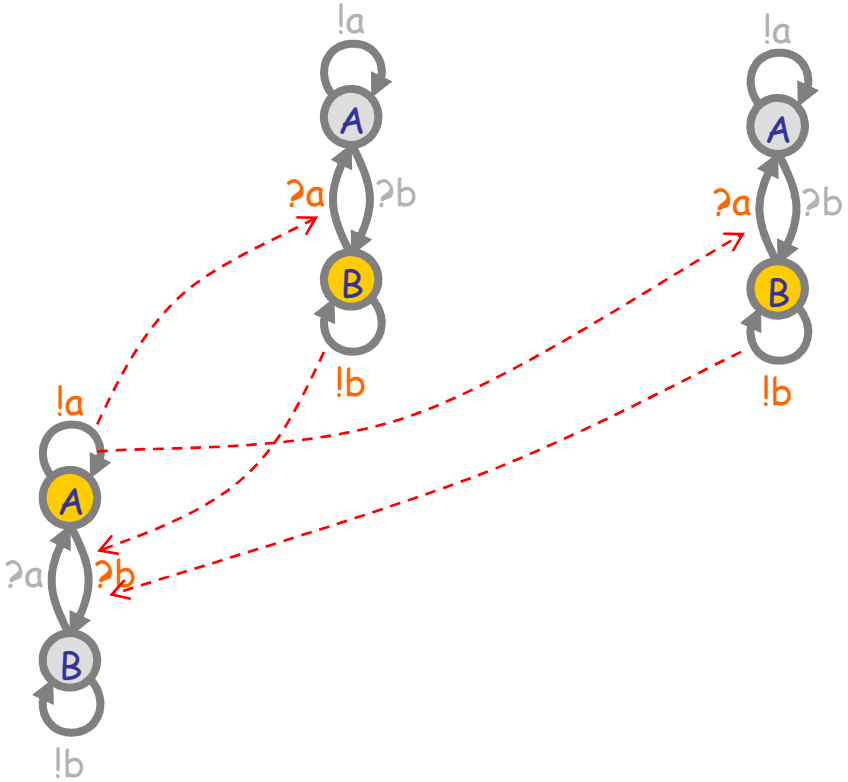


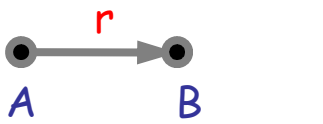


All-B stable population

Nondeterministic population behavior ("multistability")

CTMC Semantics



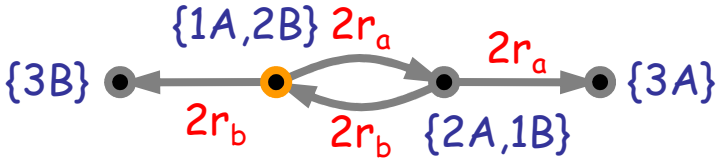


CTMC
(homogeneous) Continuous Time Markov Chain
- directed graph with no self loops
- nodes are system states
- arcs have transition rates

Probability of holding in state A:

$Pr(H_A>t) = e^{-rt}$

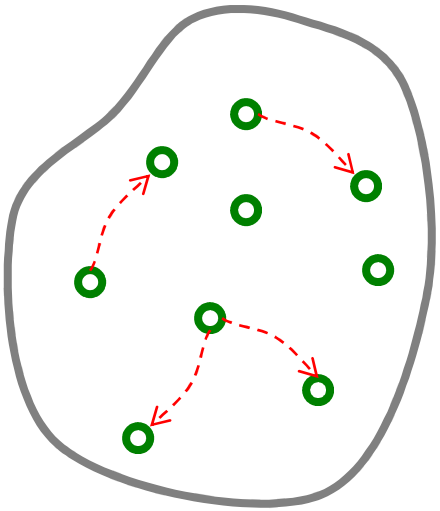
in general, $Pr(H_A>t) = e^{-Rt}$ where R is the sum of all the exit rates from A



CTMC

Stochastic Automata Collectives

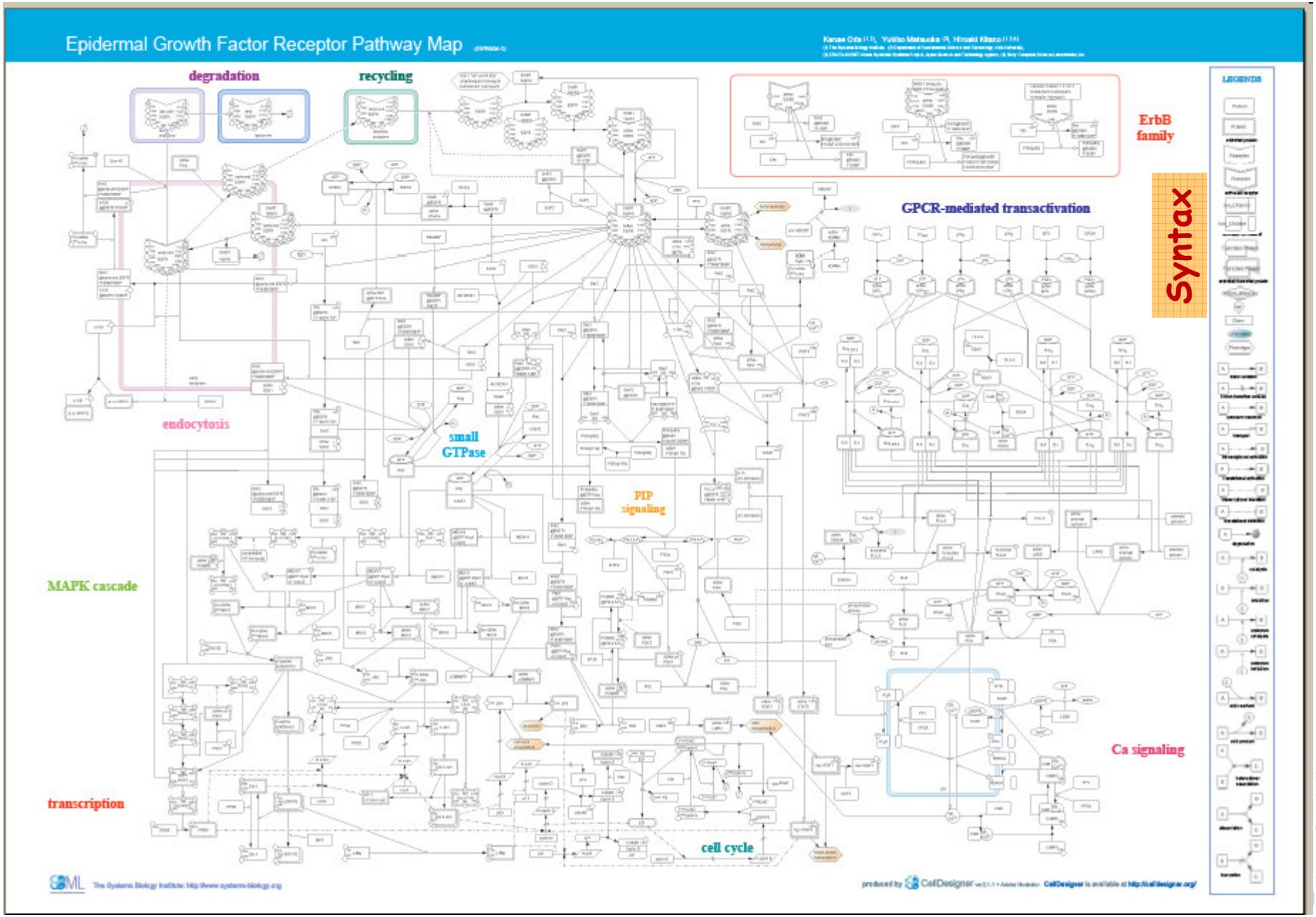
- "Collective":
  - A large set of interacting finite state automata:
    - Not quite language automata ("large set")
    - Not quite cellular automata ("interacting" but not on a grid)
    - Not quite process algebra ("collective behavior")
    - Cf. multi-agent systems and swarm intelligence



- "Stochastic":
  - Interactions have *rates*
    - Not quite discrete (hundreds or thousands of components)
    - Not quite continuous (non-trivial stochastic effects)
    - Not quite hybrid (no "switching" between regimes)

- Very much like biochemistry
  - Which is a large set of stochastically interacting molecules/proteins
  - Are proteins finite state and subject to automata-like transitions?
    - Let's say they are, at least because:
    - Much of the knowledge being accumulated in Systems Biology is described as state transition diagrams [Kitano].

Discrete State Transitions



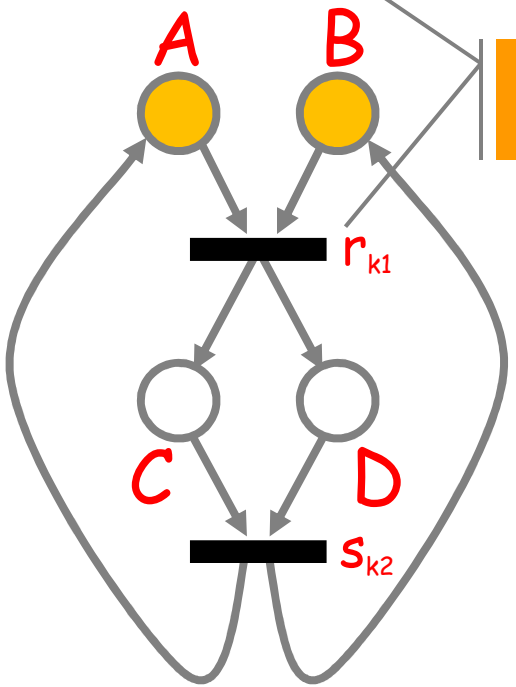
Chemistry vs. Automata

A process algebra (chemistry)

$r: A + B \rightarrow_{k1} C + D$

$s: C + D \rightarrow_{k2} A + B$

Does A become C or D?

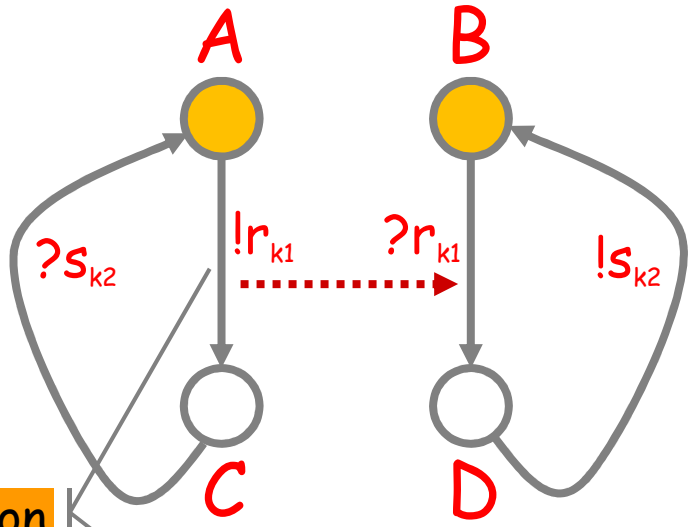


Reaction oriented

1 line per reaction

A Petri-Net-like representation. Precise and dynamic, but not modular, scalable, or maintainable.

A different process algebra (automata)



Interaction oriented

1 line per component

$A = !r_{k1}; C$

$C = ?s_{k2}; A$

$B = ?r_{k1}; D$

$D = !s_{k2}; B$

A becomes C not D!

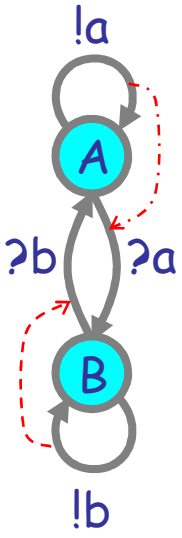
A compositional graphical representation (precise, dynamic *and* modular) and the corresponding calculus.

The same "model"

Maps to a CTMC	Maps to a CTMC

Groupies and Celebrities

Groupies and Celebrities



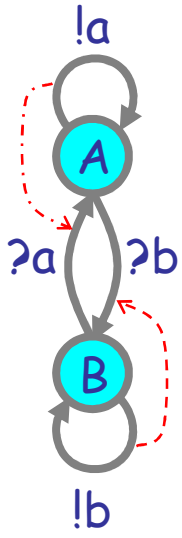
Celebrity

(does not want to be like somebody else)

```
directive sample 1.0 1000
directive plot A(); B()

new a@1.0:chan()
new b@1.0:chan()

let A() = do !a; A() or ?a; B()
and B() = do !b; B() or ?b; A()

run 100 of (A() | B())
```

a@1.0

b@1.0



Groupie

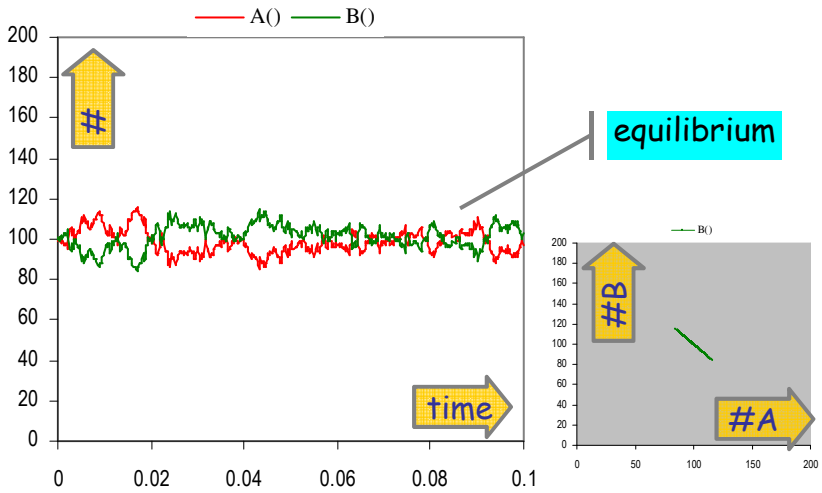
(wants to be like somebody different)

```
directive sample 1.0 1000
directive plot A(); B()

new a@1.0:chan()
new b@1.0:chan()

let A() = do !a; A() or ?b; B()
and B() = do !b; B() or ?a; A()

run 100 of (A() | B())
```

a@1.0

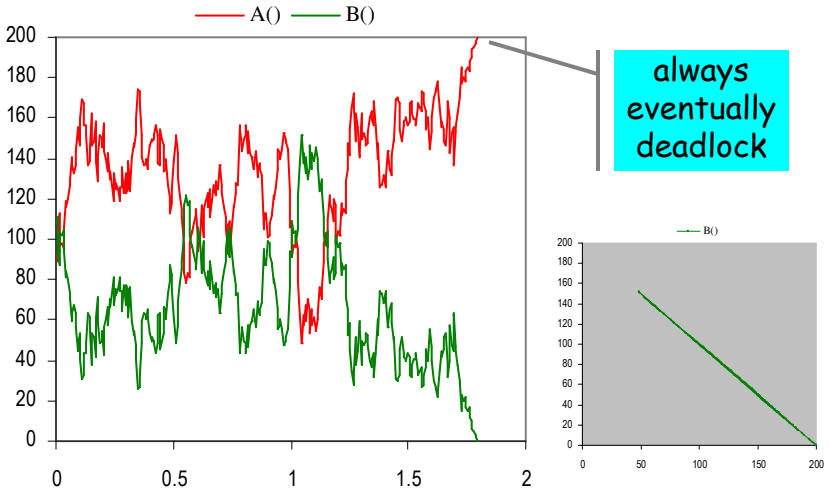
b@1.0

A stochastic collective of celebrities:



Stable because as soon as a A finds itself in the majority, it is more likely to find somebody in the same state, and hence change, so the majority is weakened.

A stochastic collective of groupies:

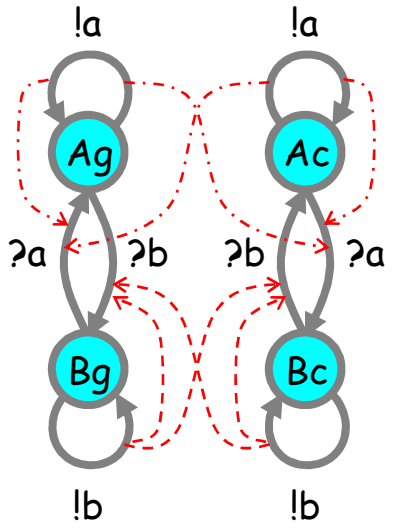


Unstable because within an A majority, an A has difficulty finding a B to emulate, but the few B's have plenty of A's to emulate, so the majority may switch to B. Leads to deadlock when everybody is in the same state and there is nobody different to emulate.

Both Together

A way to break the deadlocks: Groupies with just a few Celebrities

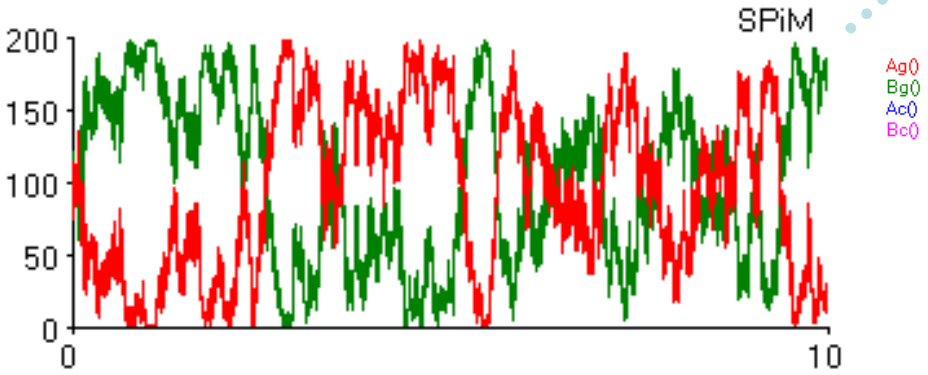
Many Groupies



A few Celebrities

```
directive sample 10.0
directive plot Ag(); Bg(); Ac(); Bc()

new a@1.0:chan()
new b@1.0:chan()

let Ac() = do !a; Ac() or ?a; Bc()
and Bc() = do !b; Bc() or ?b; Ac()

let Ag() = do !a; Ag() or ?b; Bg()
and Bg() = do !b; Bg() or ?a; Ag()

run 1 of Ac()
run 100 of (Ag() | Bg())
```

never deadlock

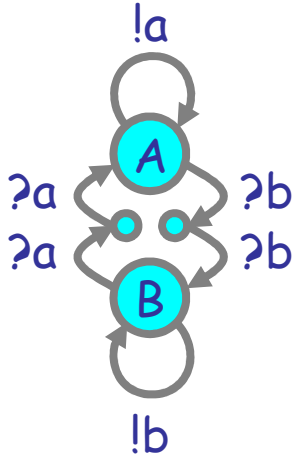


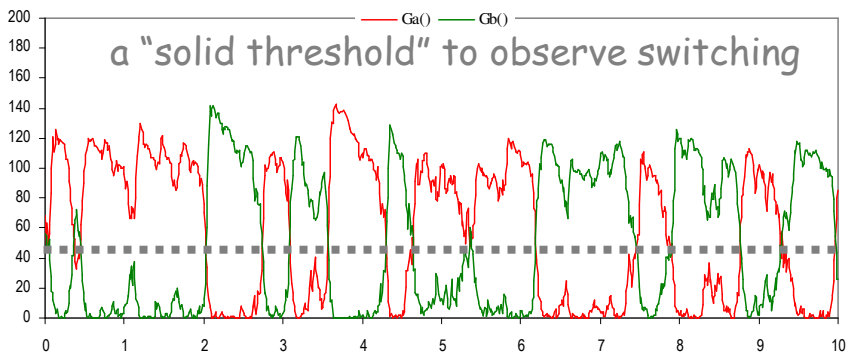
A tiny bit of "noise" can make a huge difference

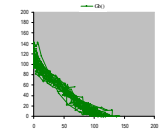
Regularity can arise not far from chaos

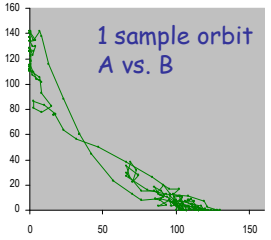
Hysteric Groupies

We can get more regular behavior from groupies if they "need more convincing", or "hysteresis" (history-dependence), to switch states.

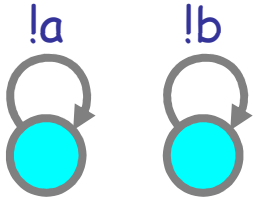








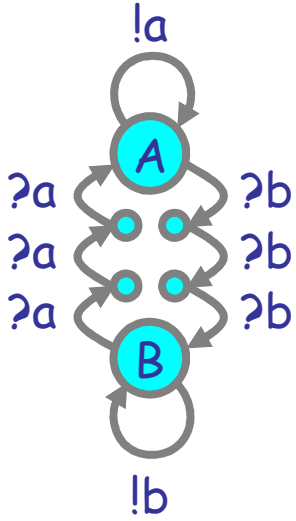

```
directive sample 10.0 1000
directive plot Ga(); Gb()

new a@1.0:chan()
new b@1.0:chan()

let Ga() = do !a; Ga() or ?b; ?b; Gb()
and Gb() = do !b; Gb() or ?a; ?a; Ga()

let Da() = !a; Da()
and Db() = !b; Db()

run 100 of (Ga() | Gb())
run   1 of (Da() | Db())
```

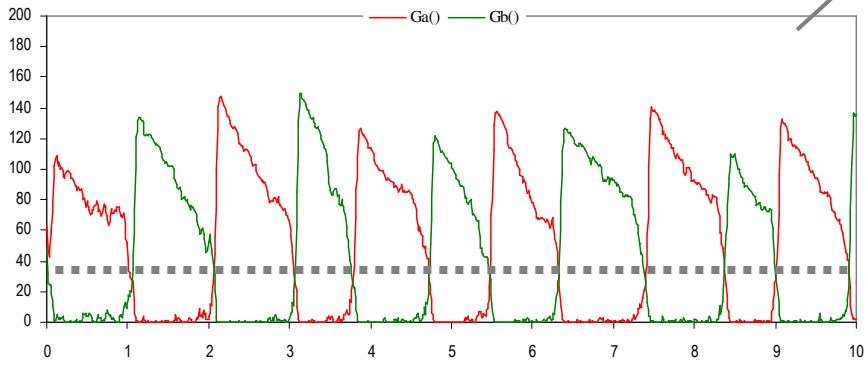


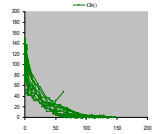
(With doping to break deadlocks)

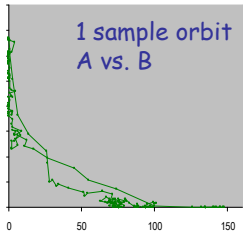
N.B.: It will not oscillate without doping (noise)

"regular" oscillation





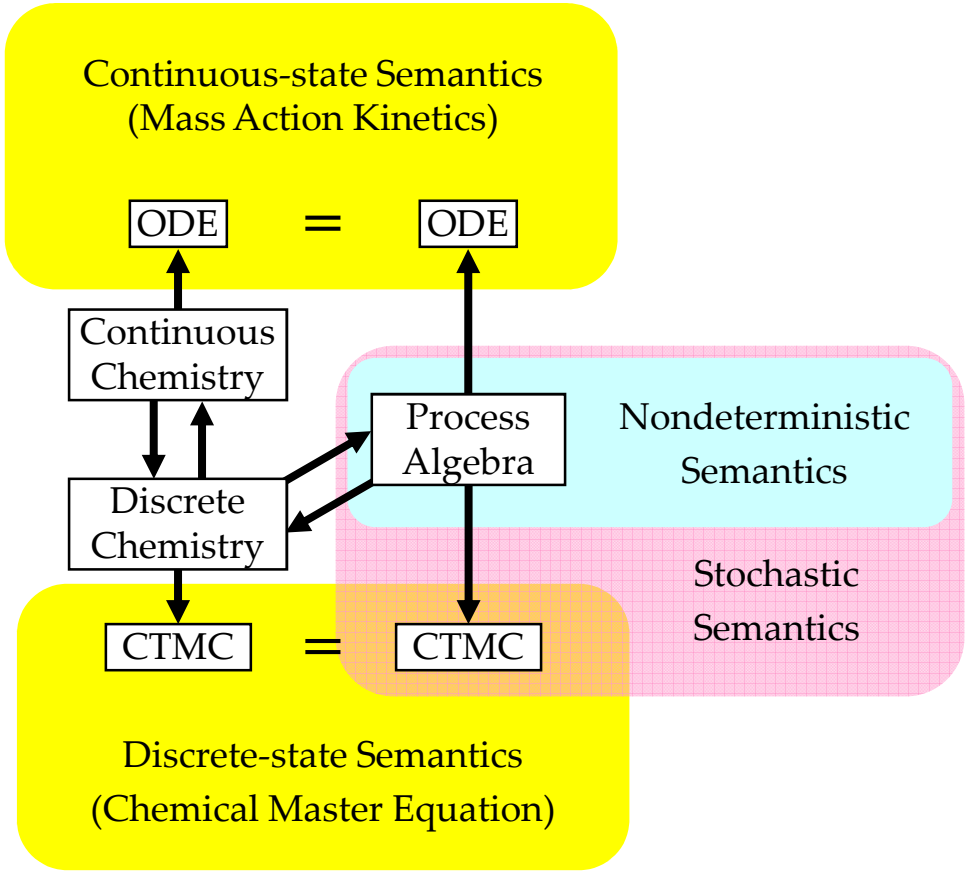





```
directive sample 10.0 1000
directive plot Ga(); Gb()

new a@1.0:chan()
new b@1.0:chan()

let Ga() = do !a; Ga() or ?b; ?b; ?b; Gb()
and Gb() = do !b; Gb() or ?a; ?a; ?a; Ga()

let Da() = !a; Da()
and Db() = !b; Db()

run 100 of (Ga() | Gb())
run   1 of (Da() | Db())
```

Semantics of Collective Behavior

The Two Semantic Sides of Chemistry

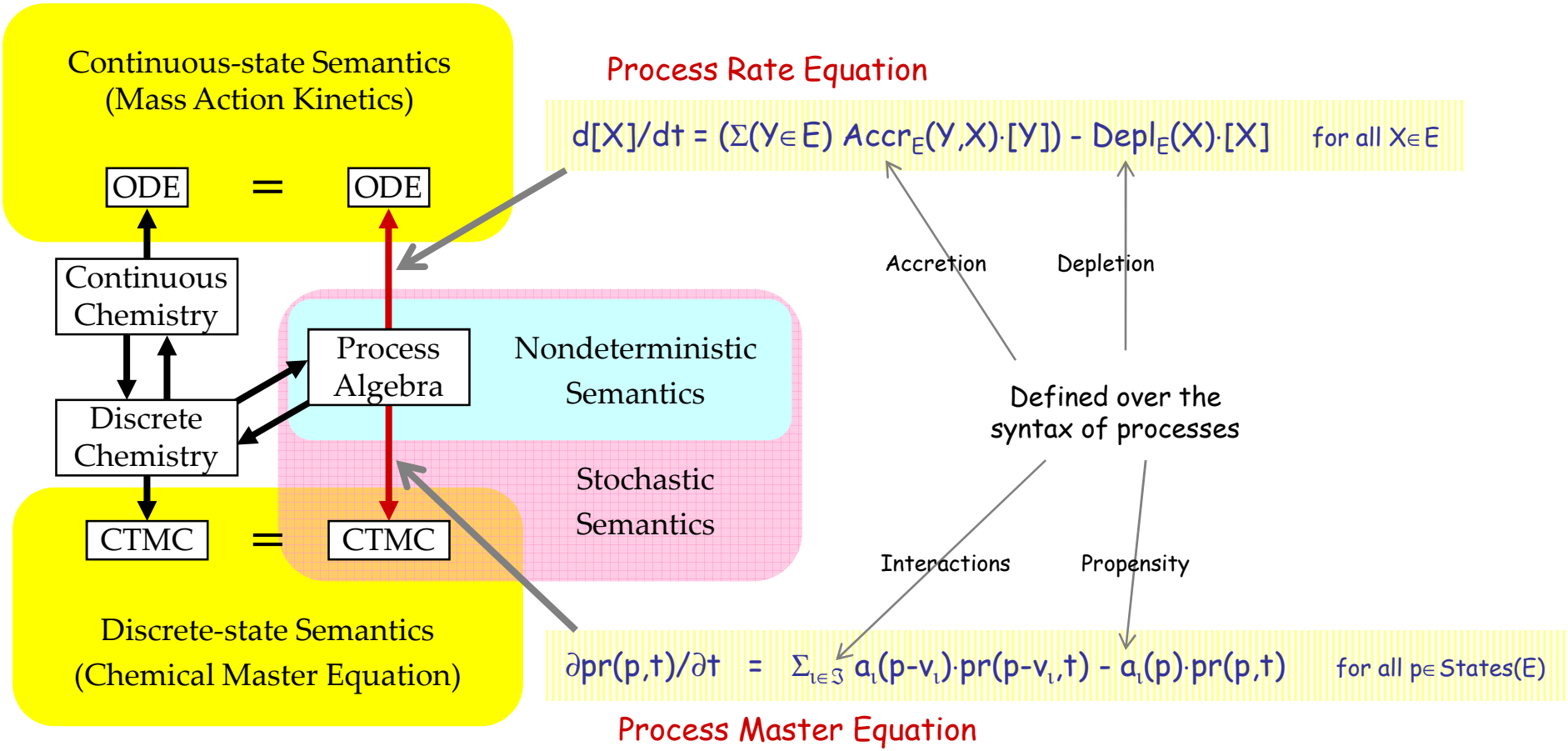


These diagrams commute via appropriate maps.

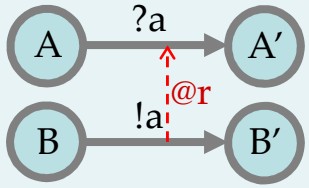
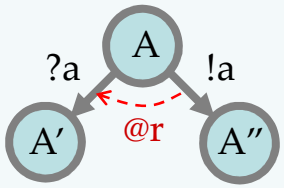
L. Cardelli: "On Process Rate Semantics" (TCS)

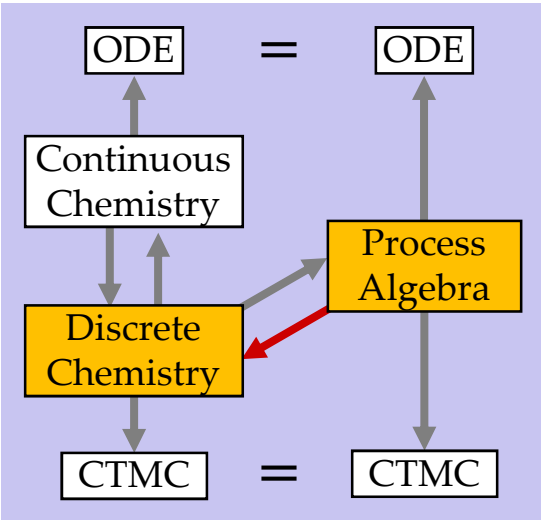
L. Cardelli: "A Process Algebra Master Equation" (QEST'07)

Quantitative Process Semantics



From Automata to Reactions (by example)

Interacting Automata →	Discrete Chemistry
initial states A | A | ... | A	initial quantities $\#A_0$
	$A \overset{r}{\dashrightarrow} A'$
	$A+B \overset{r}{\dashrightarrow} A'+B'$
	$A+A \overset{2r}{\dashrightarrow} A'+A''$



From Reactions to Automata (by example)

v_1: $A+B \to_{k_1} C+C$

v_2: $A+C \to_{k_2} D$

v_3: $C \to_{k_3} E+F$

v_4: $F+F \to_{k_4} B$

Interaction Matrix

channels and rates
(1 per reaction)

Half-rate for homeo reactions

definitions
(1 per species)

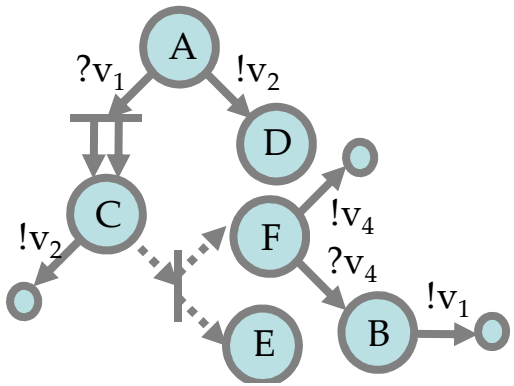
	$v_{1(k1)}$	$v_{2(k2)}$	$v_{3(k3)}$	$v_{4(k4/2)}$
A	?;(C|C)	?;D		
B	!;0			
C		!;0	τ;(E|F)	
D				
E				
F				?;B !;0

1: Fill the matrix by columns:

Degradation reaction v_i: $X \to_{k_i} P_i$
add $\tau;P_i$ to $\langle X,v_i \rangle$.

Hetero reaction v_i: $X+Y \to_{k_i} P_i$
add $?;P_i$ to $\langle X,v_i \rangle$ and $!;0$ to $\langle Y,v_i \rangle$

Homeo reaction v_i: $X+X \to_{k_i} P_i$
add $?;P_i$ and $!;0$ to $\langle X,v_i \rangle$



2: Read the result by rows:

$A = ?v_{1(k1)};(C|C) \oplus ?v_{2(k2)};D$

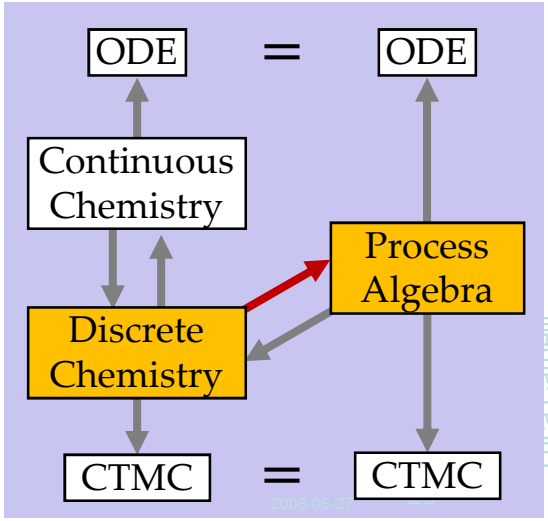
$B = !v_{1(k1)};0$

$C = !v_{2(k2)};0 \oplus \tau_{k3};(E|F)$

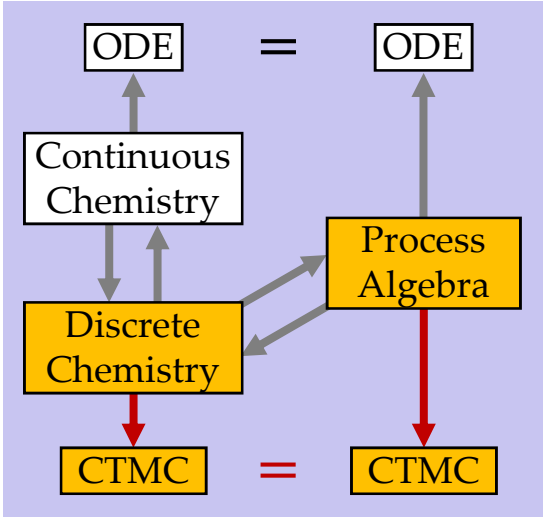
$D = 0$

$E = 0$

$F = ?v_{4(k4/2)};B \oplus !v_{4(k4/2)};0$



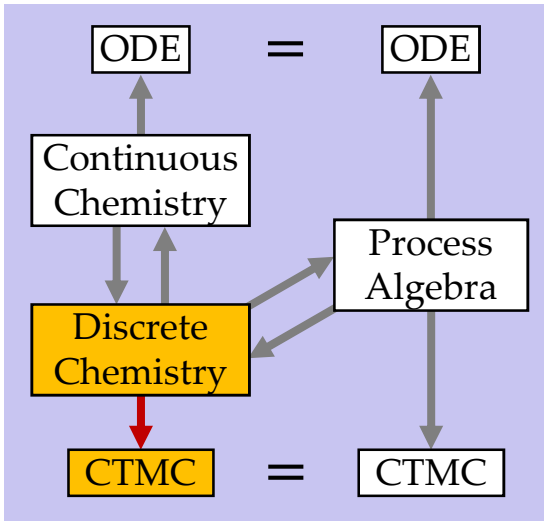
Discrete-State Semantics



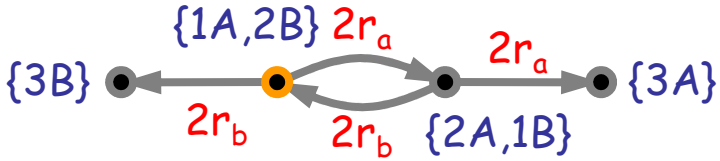
Discrete Semantics of Reactions

Syntax:

$$A+B \rightarrow^{r} A+A$$
$$A+B \rightarrow^{r} B+B$$
$$A+B+B$$

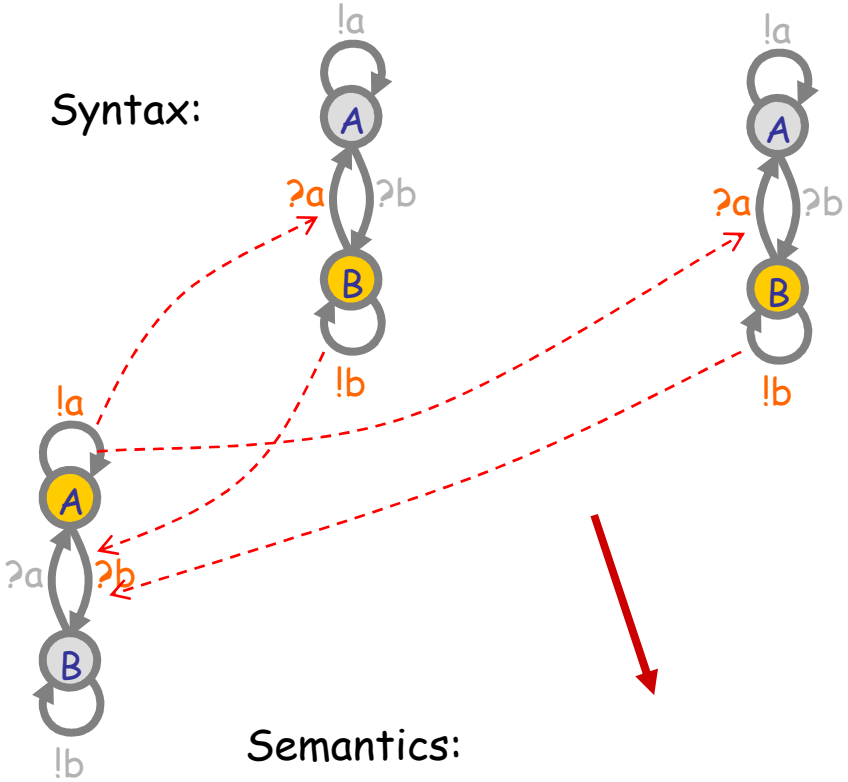


Semantics:

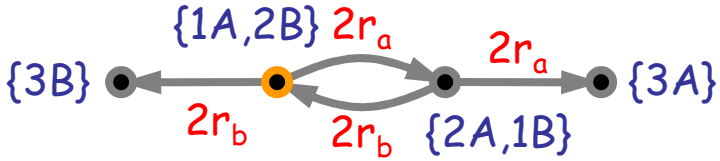


CTMC

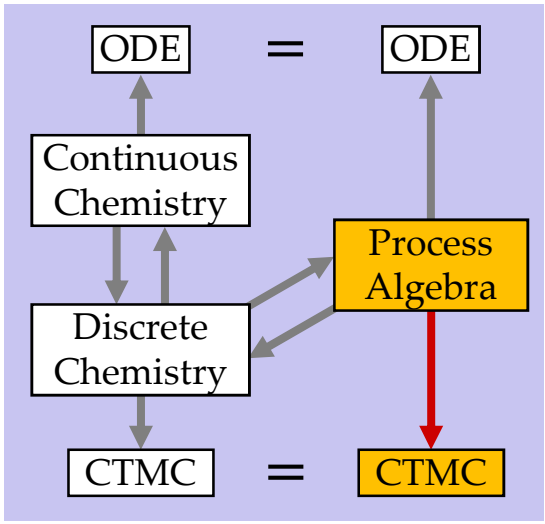
Discrete Semantics of Reagents



Semantics:

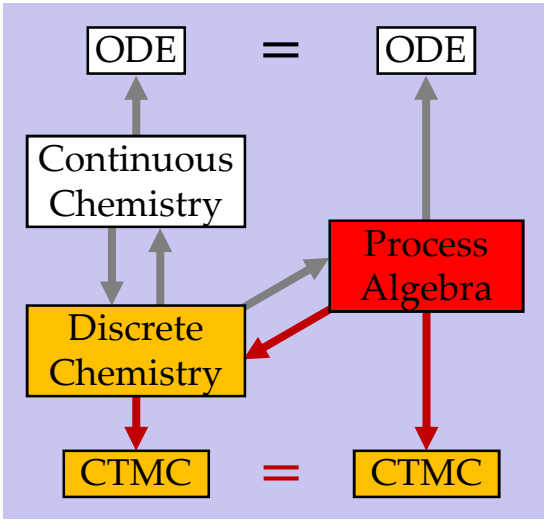


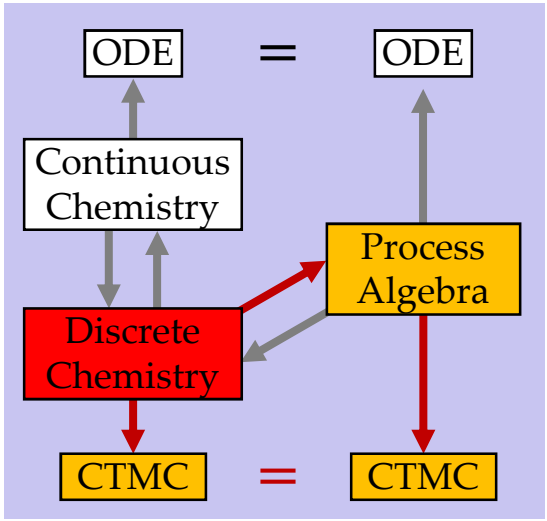
CTMC



Discrete State Equivalence

- Def: ≋ is equivalent CTMC's (isomorphic graphs with same rates).
- Thm: E ≋ Ch(E)
- Thm: C ≋ Pi(C)





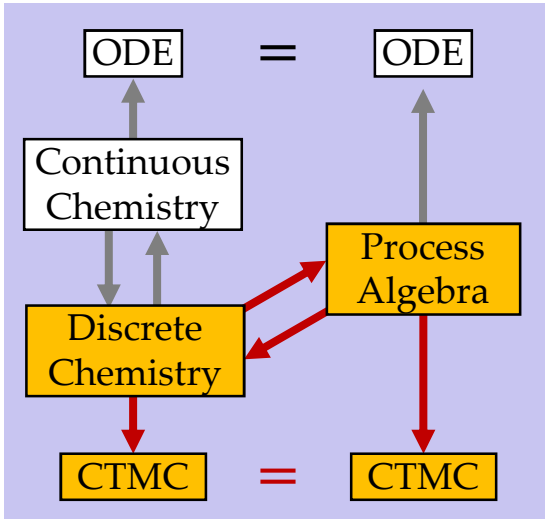
- For each E there is an E' ≋ E that is detangled (E' = Pi(Ch(E)))
- For each E in automata form there is an an E' ≋ E that is detangled and in automata form (E' = Detangle(E)).

Process Algebra = Discrete Chemistry

This is enough to establish that the process algebra is really faithful to the chemistry.

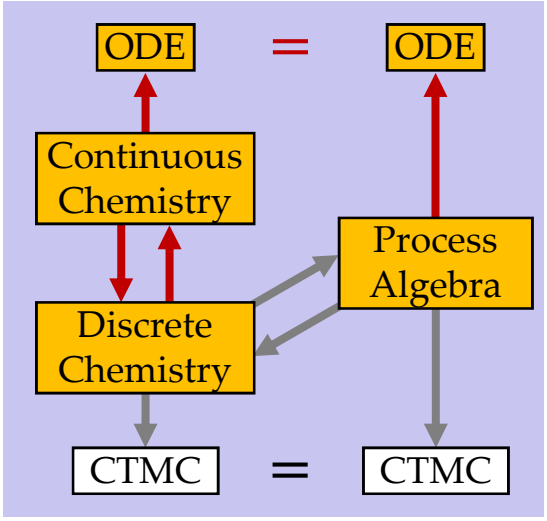
But CTMC are not the "ultimate semantics" because there are still questions of when two different CTMCs are actually equivalent (e.g. "lumping").

The "ultimate semantics" of chemistry is the *Chemical Master Equation* (derivable from the Chapman-Kolmogorov equation of the CTMC).



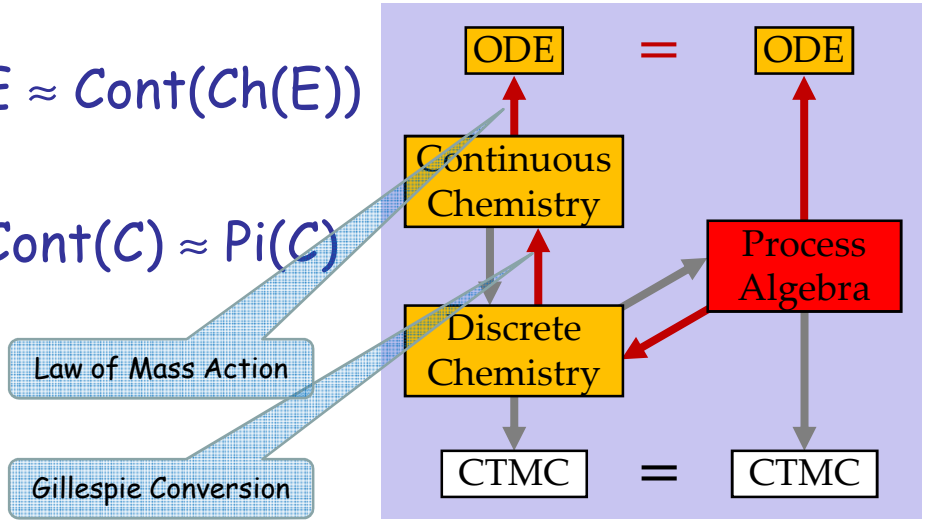
Continuous-State Semantics

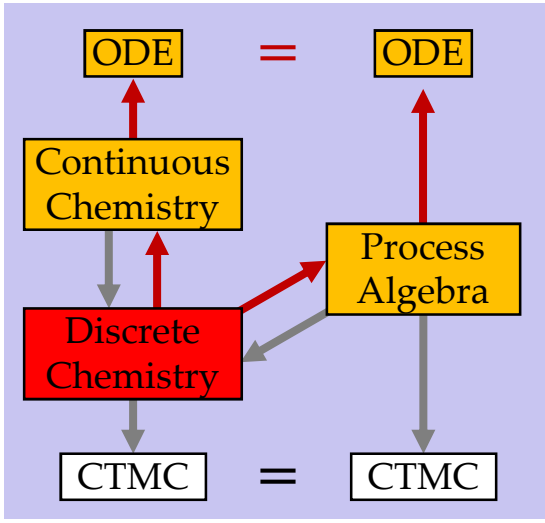
(summary)



Continuous State Equivalence

- Def: ≈ is equivalence of polynomials over the field of reals.
- Thm: E ≈ Cont(Ch(E))
- Thm: Cont(C) ≈ Pi(C)

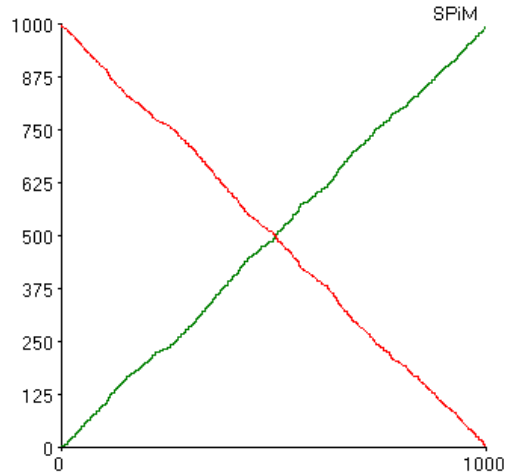




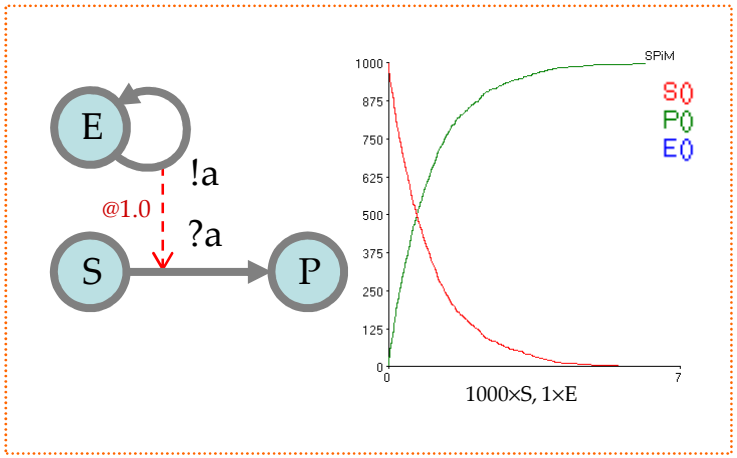
- For each E there is an E' ≈ E that is detangled (E' = Pi(Ch(E)))
- For each E in automata form there is an an E' ≈ E that is detangled and in automata form (E' = Detangle(E)).

Design Exercise: Making Lines

Build me a population like this:



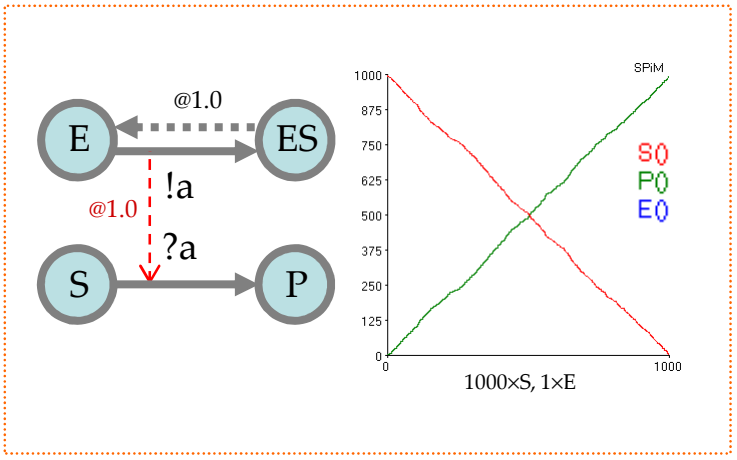
Second-order and Zero-order Regime



$E+S \rightarrow^{r} E+P$

```
directive sample 1000.0
directive plot S(); P(); E()

new a@1.0:chan()

let E() = !a; E()
and S() = ?a; P()
and P() = ()

run (1 of  E() | 1000 of S())
```

Second-Order Regime
$d[S]/dt = -r[E][S]$

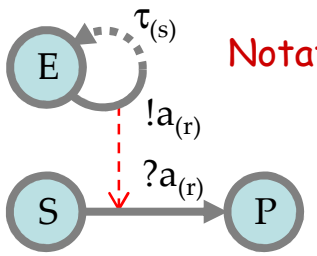


$E+S \rightarrow^{r} ES+P$

$ES \rightarrow^{s} E$

```
directive sample 1000.0
directive plot S(); P(); E()

new a@1.0:chan()

let E() = !a; delay@1.0; E()
and S() = ?a; P()
and P() = ()

run (1 of  E() | 1000 of S())
```

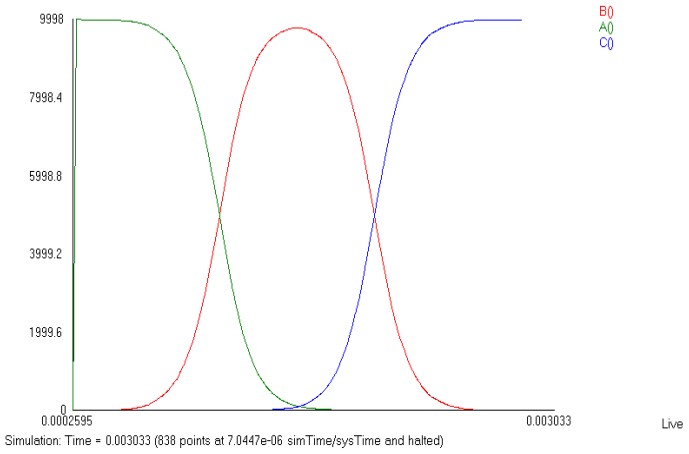
Zero-Order Regime
$d[S]/dt \cong -1$ (by assuming $d[ES]/dt = 0$)



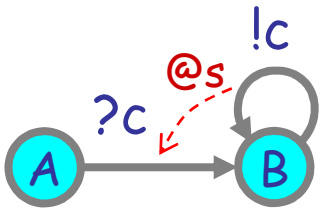
Notation

Design Exercise: Making Waves

Build me a population like this:



Nonlinear Transition (NLT)



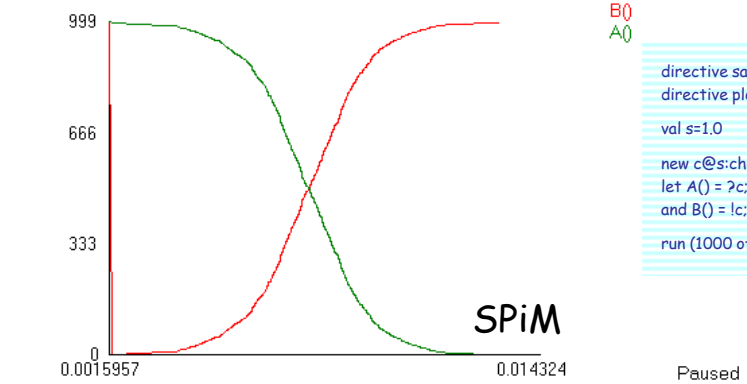
$A = ?c_{(s)};B$

$B = !c_{(s)};B$

$A+B \rightarrow^{s} B+B$

$[A]^{\bullet} = -s[A][B]$

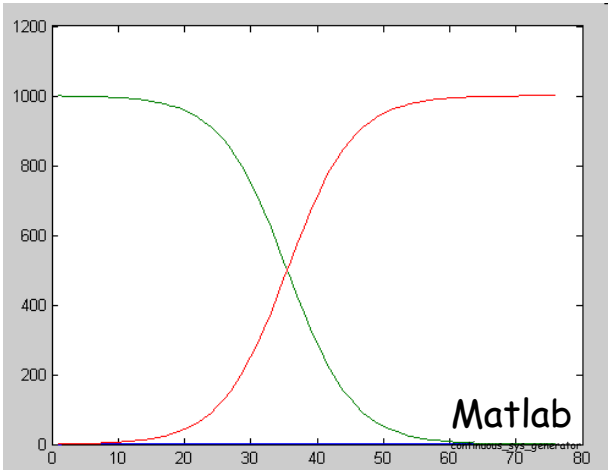
$[B]^{\bullet} = s[A][B]$



Simulation: Time = 0.013448 (999 points at 0.0085215 simTime/sysTime and halted)

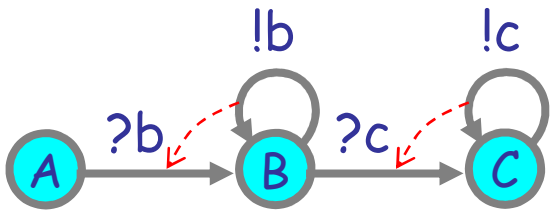
```
directive sample 0.02 1000
directive plot B(); A()

val s=1.0

new c@s:chan
let A() = ?c; B()
and B() = !c;B()

run (1000 of A() | 1 of B())
```

N.B.: needs at least 1 B to "get started".




```
interval/step [0:0.001:0.0]
(A)     dx1/dt = - x1*x2        1000.0
(B)     dx2/dt = x1*x2          1.0
```

Two NLTs: Bell Shape



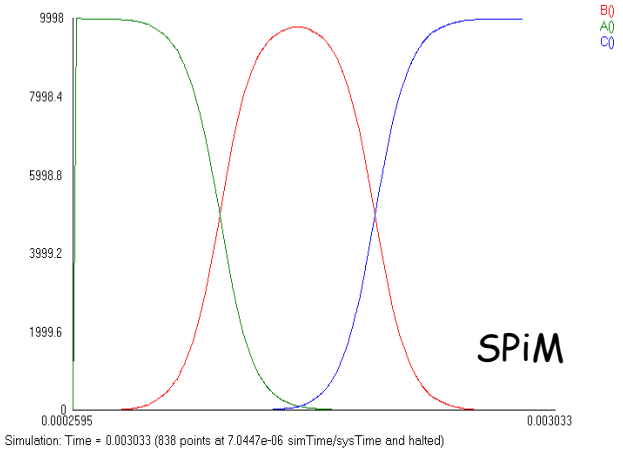
$[B]^\bullet = [B]([A]-[C])$

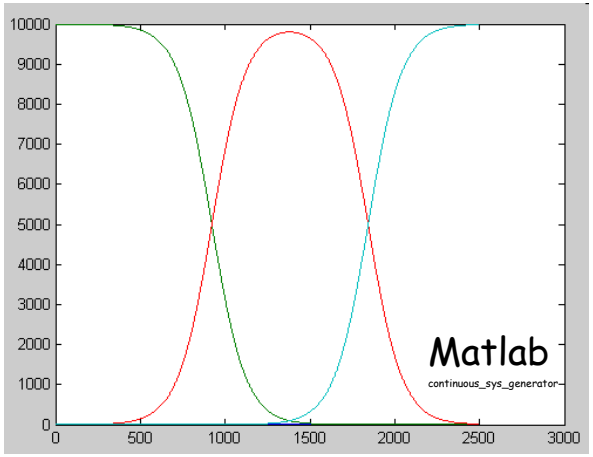
$A = ?b_{(1)};B$
$B = !b_{(1)};B \oplus ?c_{(1)};C$
$C = !c_{(1)};C$

$A+B \rightarrow^1 B+B$
$B+C \rightarrow^1 C+C$

$[A]^\bullet = -[A][B]$
$[B]^\bullet = [A][B]-[B][C]$
$[C]^\bullet = [B][C]$

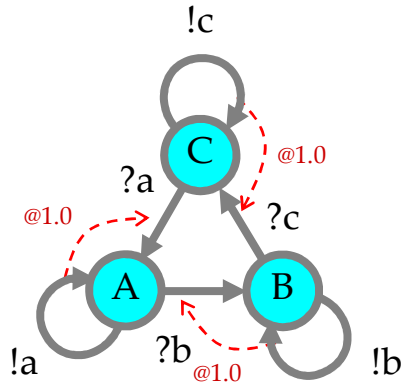
```
directive sample 0.0025 1000
directive plot B(); A(); C()

new b@1.0:chan new c@1.0:chan

let A() = ?b; B()
and B() = do !b;B() or ?c; C()
and C() = !c;C()

run ((10000 of A()) | B() | C())
```

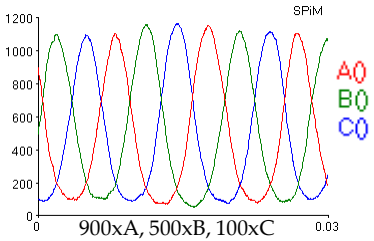





```
interval/step [0:0.000001:0.0025]
(A)   dx1/dt = -x1*x2          10000.0
(B)   dx2/dt = x1*x2 - x2*x3   1.0
(C)   dx3/dt = x2*x3           1.0
```

NLT in a Cycle: Oscillator (unstable)



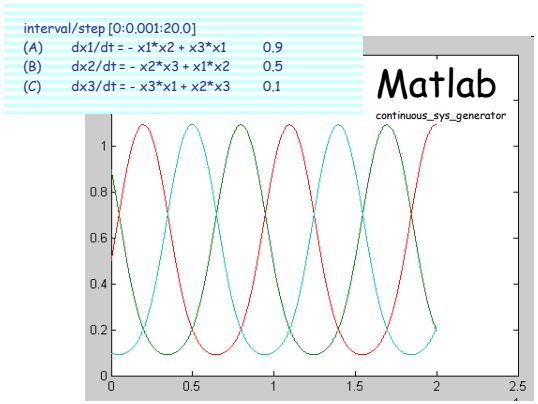


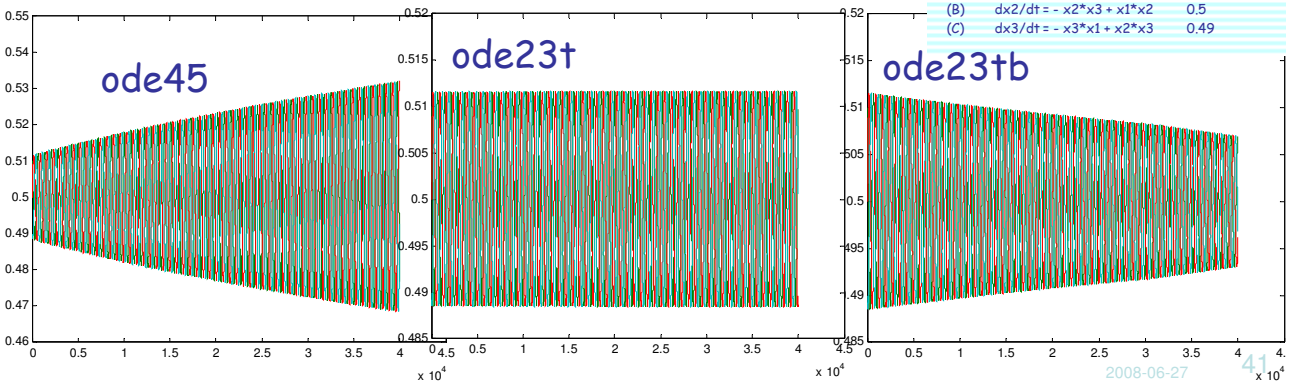

```
directive sample 0.03 1000
directive plot A(); B(); C()

new a@1.0:chan new b@1.0:chan new c@1.0:chan
let A() = do !a;A() or ?b; B()
and B() = do !b;B() or ?c; C()
and C() = do !c;C() or ?a; A()

run (900 of A() | 500 of B() | 100 of C())
```

$A = !a_{(s)};A \oplus ?b_{(s)};B$
$B = !b_{(s)};B \oplus ?c_{(s)};C$
$C = !c_{(s)};C \oplus ?a_{(s)};A$

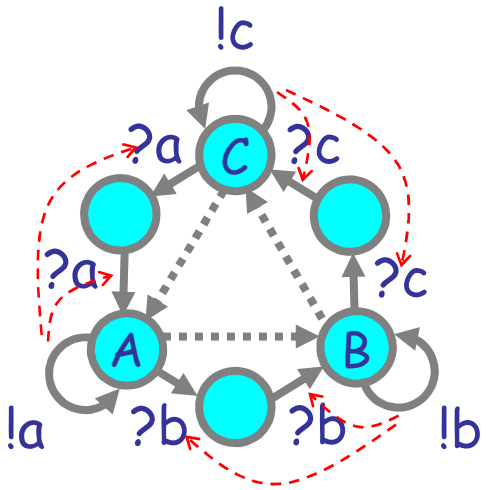
$A+B \rightarrow^{s} B+B$
$B+C \rightarrow^{s} C+C$
$C+A \rightarrow^{s} A+A$

$[A]^{\bullet} = -s[A][B]+s[C][A]$
$[B]^{\bullet} = -s[B][C]+s[A][B]$
$[C]^{\bullet} = -s[C][A]+s[B][C]$





Oscillator (stable)



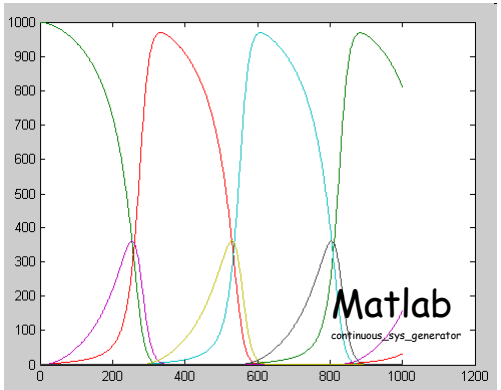

```
directive sample 0.1 1000
directive plot A1(); A2(); A3()

val r=1.0 val s=1.0

new a1@s:chan new a2@s:chan new a3@s:chan
let A1() = do !a1;A1() or delay@r;A2() or ?a2; ?a2; A2()
and A2() = do !a2;A2() or delay@r;A3() or ?a3; ?a3; A3()
and A3() = do !a3;A3() or delay@r;A1() or ?a1; ?a1; A1()

run 1000 of A1()
```

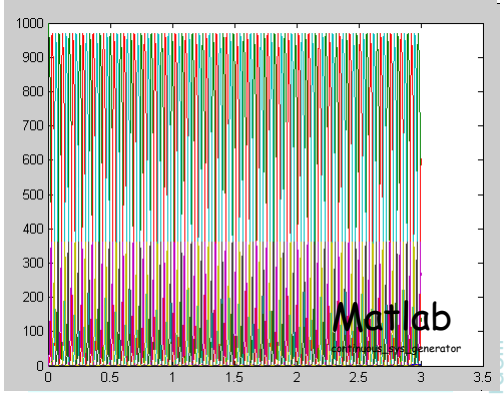
N.B. this does not deadlock!

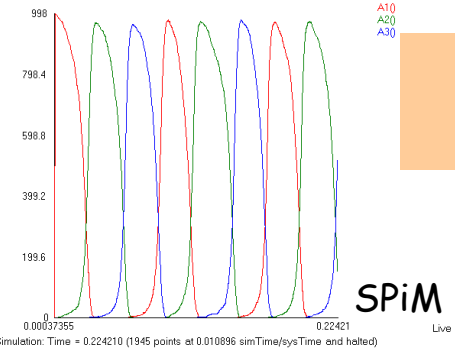
$$A = !a_{(s)};A \oplus \tau_r;B \oplus ?b_{(s)};A'$$
$$A' = ?b_{(s)};B$$
$$B = !b_{(s)};B \oplus \tau_r;C \oplus ?c_{(s)};B'$$
$$B' = ?c_{(s)};C$$
$$C = !c_{(s)};C \oplus \tau_r;A \oplus ?a_{(s)};C'$$
$$C' = ?a_{(s)};A$$

Sustained Deterministic Oscillation

$$A \rightarrow^r B$$
$$A+B \rightarrow^s A'+B$$
$$A'+B \rightarrow^s B+B$$
$$B \rightarrow^r C$$
$$B+C \rightarrow^s B'+C$$
$$B'+C \rightarrow^s C+C$$
$$C \rightarrow^r A$$
$$C+A \rightarrow^s C'+A$$
$$C'+A \rightarrow^s A+A$$

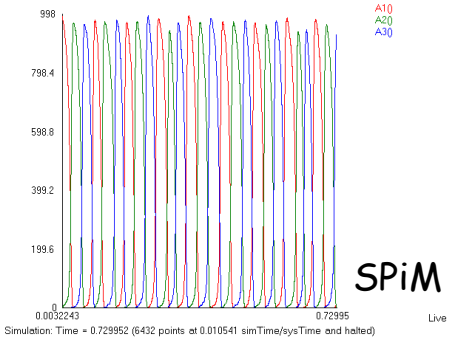






Simulation: Time = 0.224210 (1945 points at 0.010896 simTime/sysTime and halted)

Robust Stochastic Oscillation

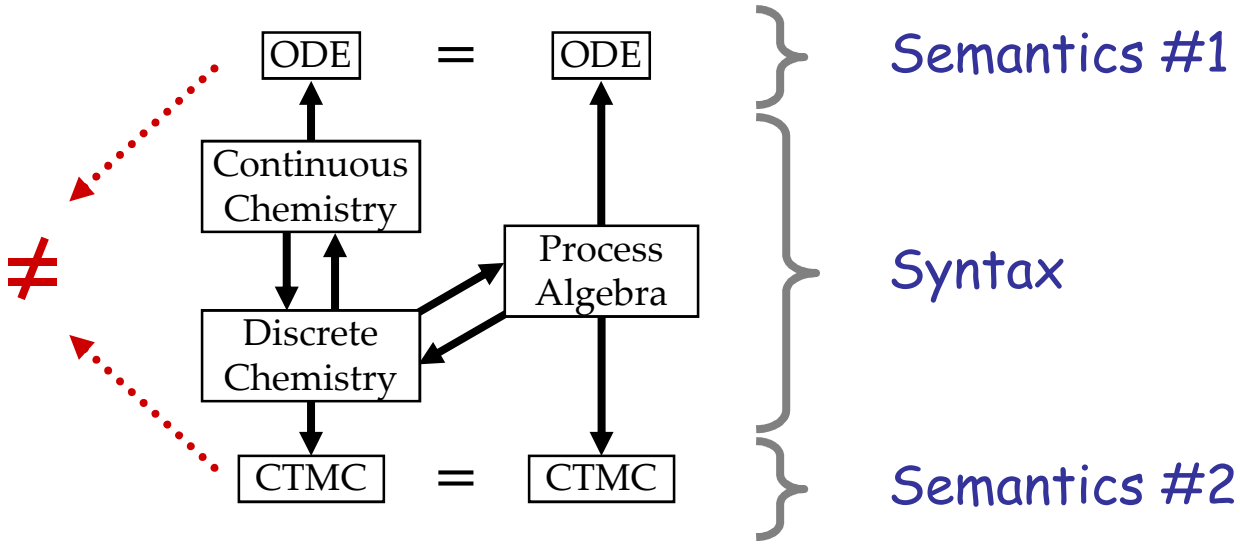


Simulation: Time = 0.729952 (6432 points at 0.010541 simTime/sysTime and halted)

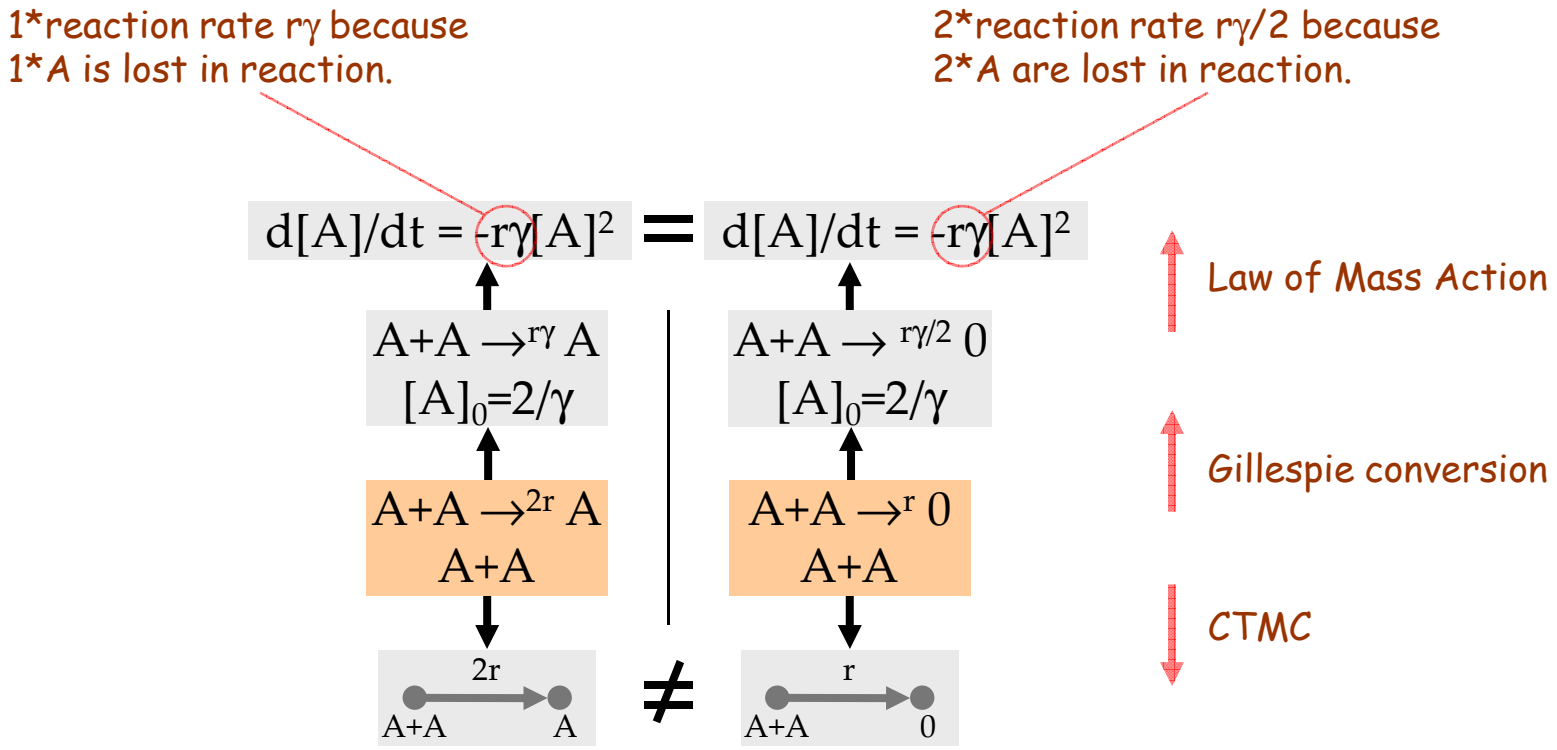
$$[A]^\bullet = -r[A]-s[A][B]+r[C]+s[C'][A]$$
$$[B]^\bullet = -r[B]-s[B][C]+r[A]+s[A'][B]$$
$$[C]^\bullet = -r[C]-s[C][A]+r[B]+s[B'][C]$$
$$[A']^\bullet = -s[A'][B] + s[A][B]$$
$$[B']^\bullet = -s[B'][C] + s[B][C]$$
$$[C']^\bullet = -s[C'][A] + s[C][A]$$

```
interval/step [0:0.0001:0.1]
(A)   dx1/dt = -x1 - x1*x2 + x3 + x6*x1    1000.0
(B)   dx2/dt = -x2 - x2*x3 + x1 + x4*x2    0.0
(C)   dx3/dt = -x3 - x3*x1 + x2 + x5*x3    0.0
(A')  dx4/dt = -x4*x2 + x1*x2              0.0
(B')  dx5/dt = -x5*x3 + x2*x3              0.0
(C')  dx6/dt = -x6*x1 + x3*x1              0.0
```

GMA ≠ CME

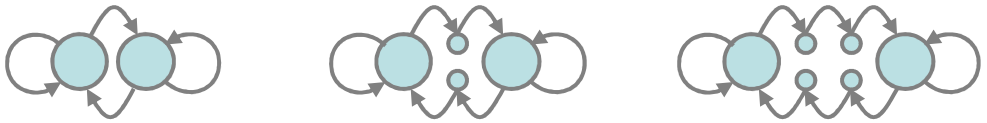

ODE
=
ODE
Continuous Chemistry
Process Algebra
Discrete Chemistry
CTMC
=
CTMC
≠
Semantics #1
Syntax
Semantics #2

$A+A \rightarrow^{2r} A$ =? $A+A \rightarrow^{r} 0$

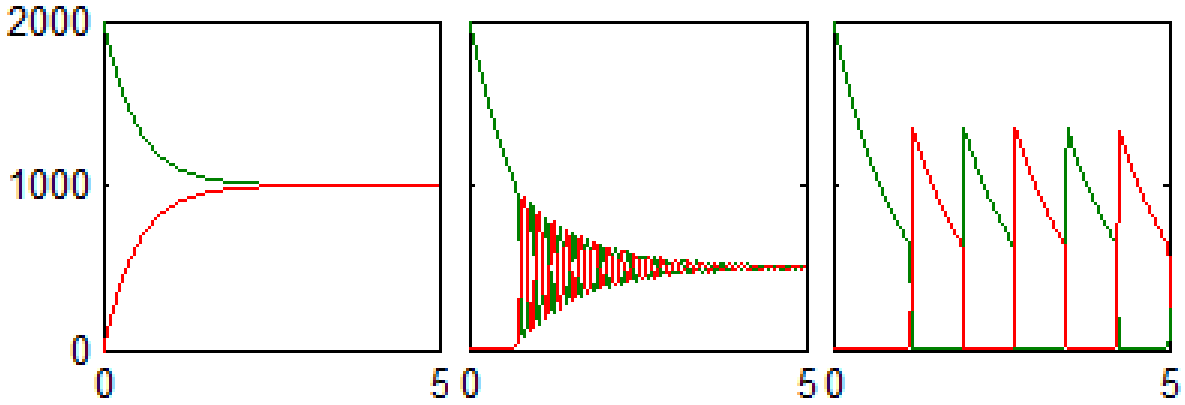


(For conservation of mass, consider instead $A+A \rightarrow^{2r} A+B$ vs. $A+A \rightarrow^{r} B+B$)

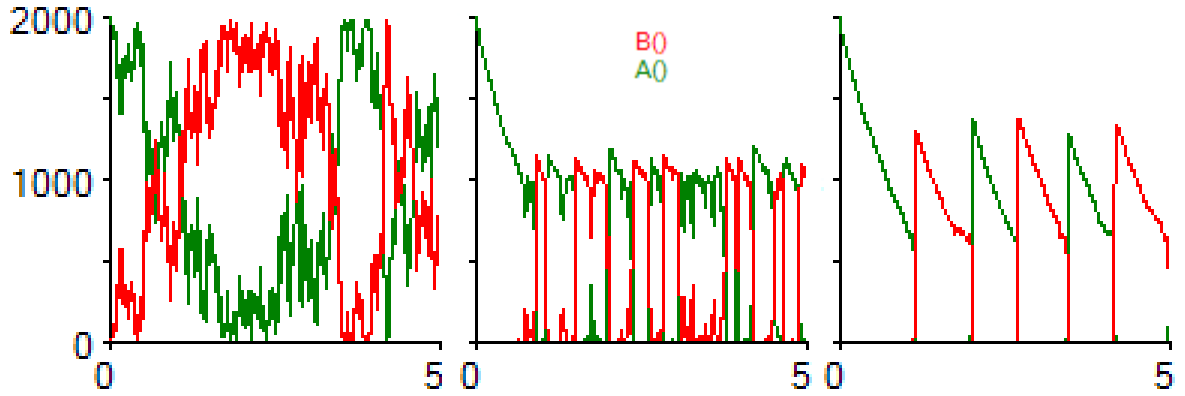
Continuous vs. Discrete Groupies



(all with doping)

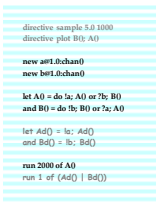


Matlab

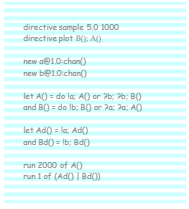


SPiM

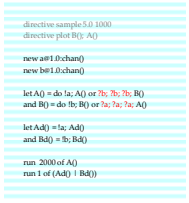
$2000\times A\ ,\ 0\times B\ ,\ 1\times A_d\ ,\ 1\times B_d\ ,\ r = 1.0$



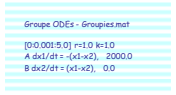
```
directive sample 5.0 1000
directive plot B(); A()

new a@1.0:chan()
new b@1.0:chan()

let A() = do !a; A() or ?b; B()
and B() = do !b; B() or ?a; A()

let Ad() = !a; Ad()
and Bd() = !b; Bd()

run 2000 of A()
run 1 of (Ad() | Bd())
```



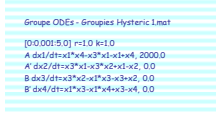
```
directive sample 5.0 1000
directive plot B(); A()

new a@1.0:chan()
new b@1.0:chan()

let A() = do !a; A() or ?b; B()
and B() = do !b; B() or ?a; A()

let Ad() = !a; Ad()
and Bd() = !b; Bd()

run 2000 of A()
run 1 of (Ad() | Bd())
```



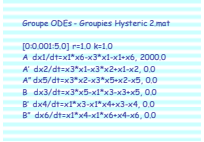
```
directive sample 5.0 1000
directive plot B(); A()

new a@1.0:chan()
new b@1.0:chan()

let A() = do !a; A() or ?b; ?b; B()
and B() = do !b; B() or ?a; ?a; A()

let Ad() = !a; Ad()
and Bd() = !b; Bd()

run 2000 of A()
run 1 of (Ad() | Bd())
```



```
Groupe ODEs - Groupies.mat

[0:0.001:5.0] r=1.0 k=1.0
A dx1/dt = -(x1-x2),   2000.0
B dx2/dt = (x1-x2),    0.0
```

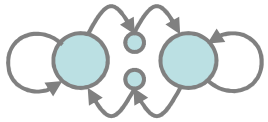


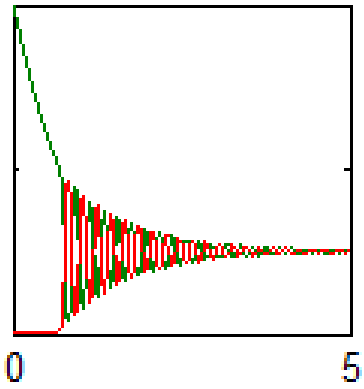
```
Groupe ODEs - Groupies Hysteric 1.mat

[0:0.001:5.0] r=1.0 k=1.0
A dx1/dt=x1*x4-x3*x1-x1+x4, 2000.0
A' dx2/dt=x3*x1-x3*x2+x1-x2, 0.0
A'' dx5/dt=x3*x2-x3*x5+x2-x5, 0.0
B dx3/dt=x3*x2-x1*x3-x3+x2, 0.0
B' dx4/dt=x1*x3-x1*x4+x3-x4, 0.0
```



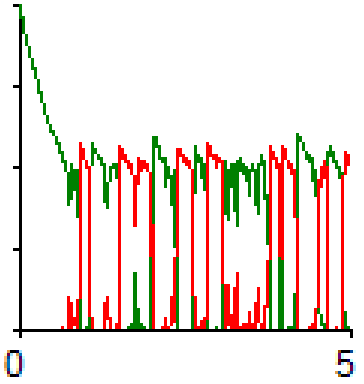
```
Groupe ODEs - Groupies Hysteric 2.mat

[0:0.001:5.0] r=1.0 k=1.0
A  dx1/dt=x1*x6-x3*x1-x1+x6,  2000.0
A'  dx2/dt=x3*x1-x3*x2+x1-x2, 0.0
A'' dx5/dt=x3*x2-x3*x5+x2-x5, 0.0
B  dx3/dt=x3*x5-x1*x3-x3+x5, 0.0
B'  dx4/dt=x1*x3-x1*x4+x3-x4, 0.0
B'' dx6/dt=x1*x4-x1*x6+x4-x6, 0.0
```

Scientific Predictions



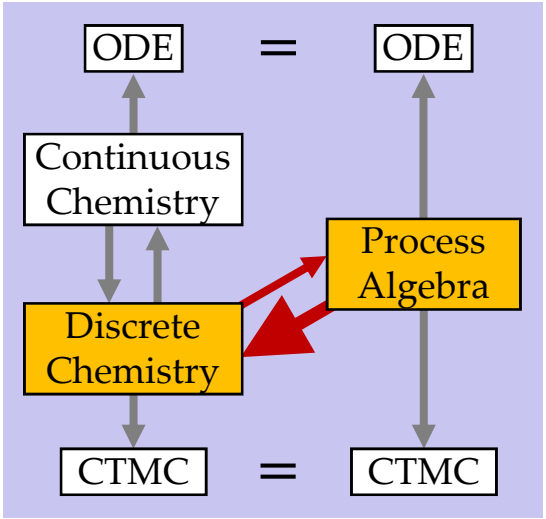


After a while, all 4 states are almost equally occupied.

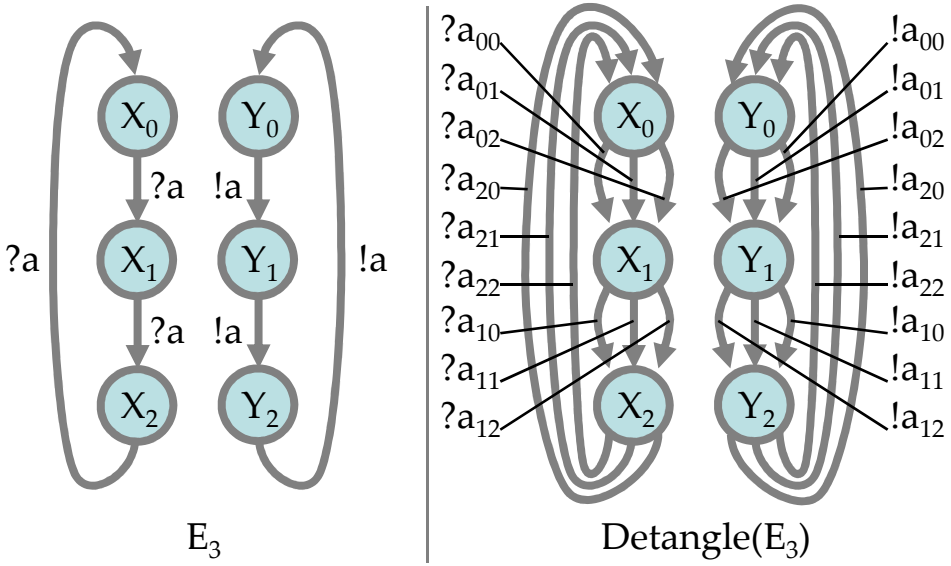


The 4 states are almost never equally occupied.

Model Compactness



Entangled vs detangled



E_3 | Detangle(E_3)

(closely related to
Pi(Ch(E_3)))

n^2 Scaling Problems

- E_n has 2n variables (nodes) and 2n terms (arcs).
- $Ch(E_n)$ has 2n species and n^2 reactions.
- The stoichiometric matrix has size $2n \cdot n^2 = 2n^3$.
- The ODEs have 2n variables and $2n(n+n) = 4n^2$ terms
  (number of variables times number of accretions plus depletions when sums are distributed)

E_3

$X_0 = ?a_{(r)};X_1$
$X_1 = ?a_{(r)};X_2$
$X_2 = ?a_{(r)};X_0$
$Y_0 = !a_{(r)};Y_1$
$Y_1 = !a_{(r)};Y_2$
$Y_2 = !a_{(r)};Y_0$

$Ch(E_3)$

$a_{00}: X_0+Y_0 \to^r X_1+Y_1$
$a_{01}: X_0+Y_1 \to^r X_1+Y_2$
$a_{02}: X_0+Y_2 \to^r X_1+Y_0$
$a_{10}: X_1+Y_0 \to^r X_2+Y_1$
$a_{11}: X_1+Y_1 \to^r X_2+Y_2$
$a_{12}: X_1+Y_2 \to^r X_2+Y_0$
$a_{20}: X_2+Y_0 \to^r X_0+Y_1$
$a_{21}: X_2+Y_1 \to^r X_0+Y_2$
$a_{22}: X_2+Y_2 \to^r X_0+Y_0$

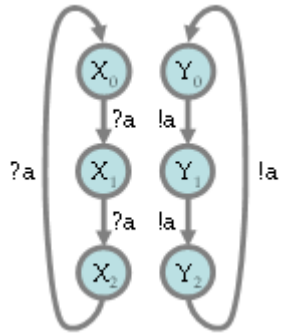
StoichiometricMatrix($Ch(E_3)$)

	a_{00}	a_{01}	a_{02}	a_{10}	a_{11}	a_{12}	a_{20}	a_{21}	a_{22}
X_0	-1	-1	-1				+1	+1	+1
X_1	+1	+1	+1	-1	-1	-1			
X_2				+1	+1	+1	-1	-1	-1
Y_0	-1		+1	-1		+1	-1		+1
Y_1	+1	-1		+1	-1		+1	-1	
Y_2		+1	-1		+1	-1		+1	-1

$ODE(E_3)$

$d[X_0]/dt = -r[X_0][Y_0] - r[X_0][Y_1] - r[X_0][Y_2] + r[X_2][Y_0] + r[X_2][Y_1] + r[X_2][Y_2]$
$d[X_1]/dt = -r[X_1][Y_0] - r[X_1][Y_1] - r[X_1][Y_2] + r[X_0][Y_0] + r[X_0][Y_1] + r[X_0][Y_2]$
$d[X_2]/dt = -r[X_2][Y_0] - r[X_2][Y_1] - r[X_2][Y_2] + r[X_1][Y_0] + r[X_1][Y_1] + r[X_1][Y_2]$
$d[Y_0]/dt = -r[X_0][Y_0] - r[X_1][Y_0] - r[X_2][Y_0] + r[X_0][Y_2] + r[X_1][Y_2] + r[X_2][Y_2]$
$d[Y_1]/dt = -r[X_0][Y_1] - r[X_1][Y_1] - r[X_2][Y_1] + r[X_0][Y_0] + r[X_1][Y_0] + r[X_2][Y_0]$
$d[Y_2]/dt = -r[X_0][Y_2] - r[X_1][Y_2] - r[X_2][Y_2] + r[X_0][Y_1] + r[X_1][Y_1] + r[X_2][Y_1]$

=



On the Computational Power of Biochemistry

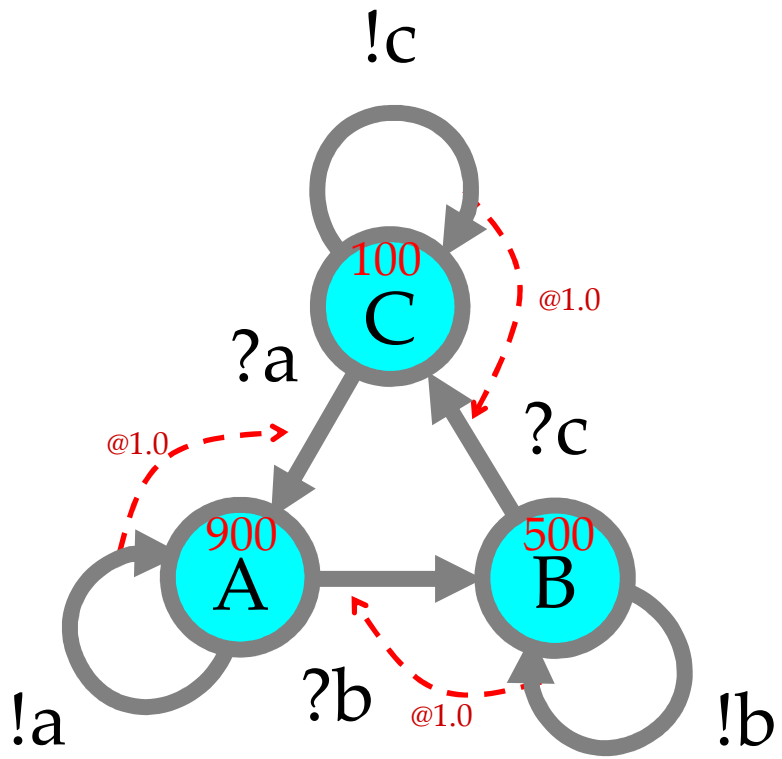
joint work with

Gianluigi Zavattaro

University of Bologna

in: Algebraic Biology '08

Can this program terminate?



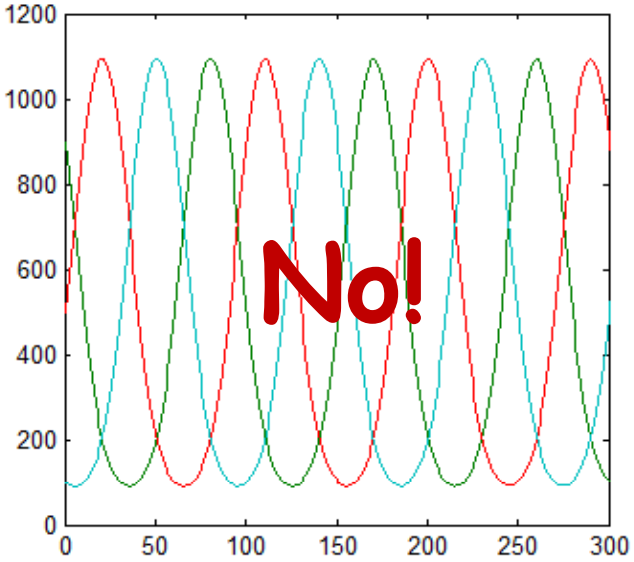
b: A+B → B+B

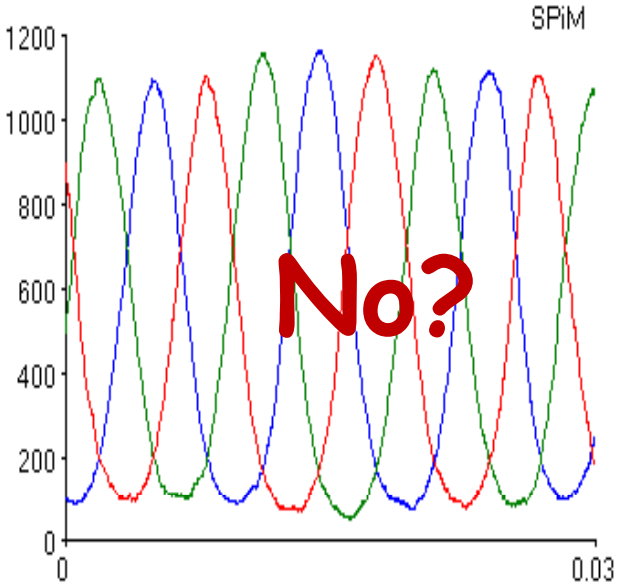
c: B+C → C+C

a: C+A → A+A

900A + 500B + 100C

"Experimantal evidence"





Continuous-State Simulation

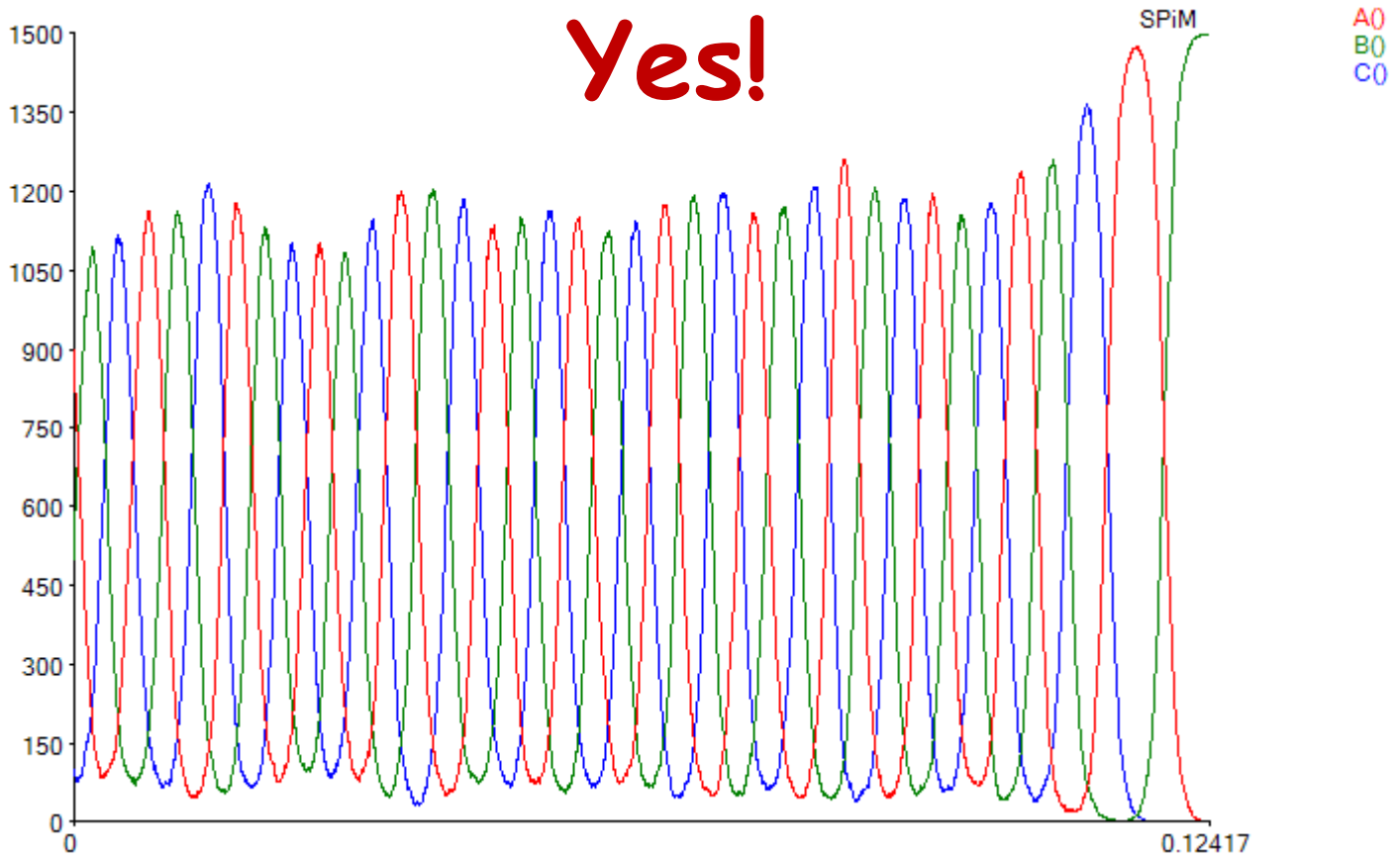
```
interval/step [0:0.0001:0.03]
(A)  dx1/dt = - x1*x2 + x3*x1    900.0
(B)  dx2/dt = - x2*x3 + x1*x2    500.0
(C)  dx3/dt = - x3*x1 + x2*x3    100.0
```

Discrete-State Simulation

```
directive sample 0.03 1000
directive plot A(); B(); C()

new a@1.0:chan new b@1.0:chan new c@1.0:chan
let A() = do !a;A() or ?b; B()
and B() = do !b;B() or ?c; C()
and C() = do !c;C() or ?a; A()

run (900 of A() | 500 of B() | 100 of C())
```

But in a longer simulation…

Yes!



Discrete-State Simulation

0A + 1500B + 0C

Is termination (possible death) decidable in Chemistry?

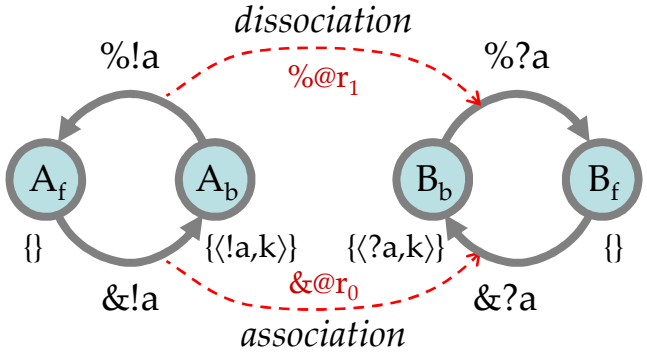
- Three equivalent definitions of "basic chemistry":
  - FSRN: Finite Stochastic Reaction Networks (finite systems of stochastic chemical reactions)
  - CGF: our process algebra.
  - Place-Transition Petri nets.

- Surprising answer: termination in basic chemistry is *decidable!*
  - (Soloveichik et al. *Computation with Finite Stochastic Chemical Reaction Networks*. In Nat. Computing. 2008) by reduction to a decidable problem in Petri Nets (reachability).

- Hence, basic chemistry cannot compute!
  - By Turing's theorem, termination for a universal computer is undecidable.
  - Hence basic chemistry is not Turing-complete.
  - (Although the full story for stochastic systems is a bit more subtle.)

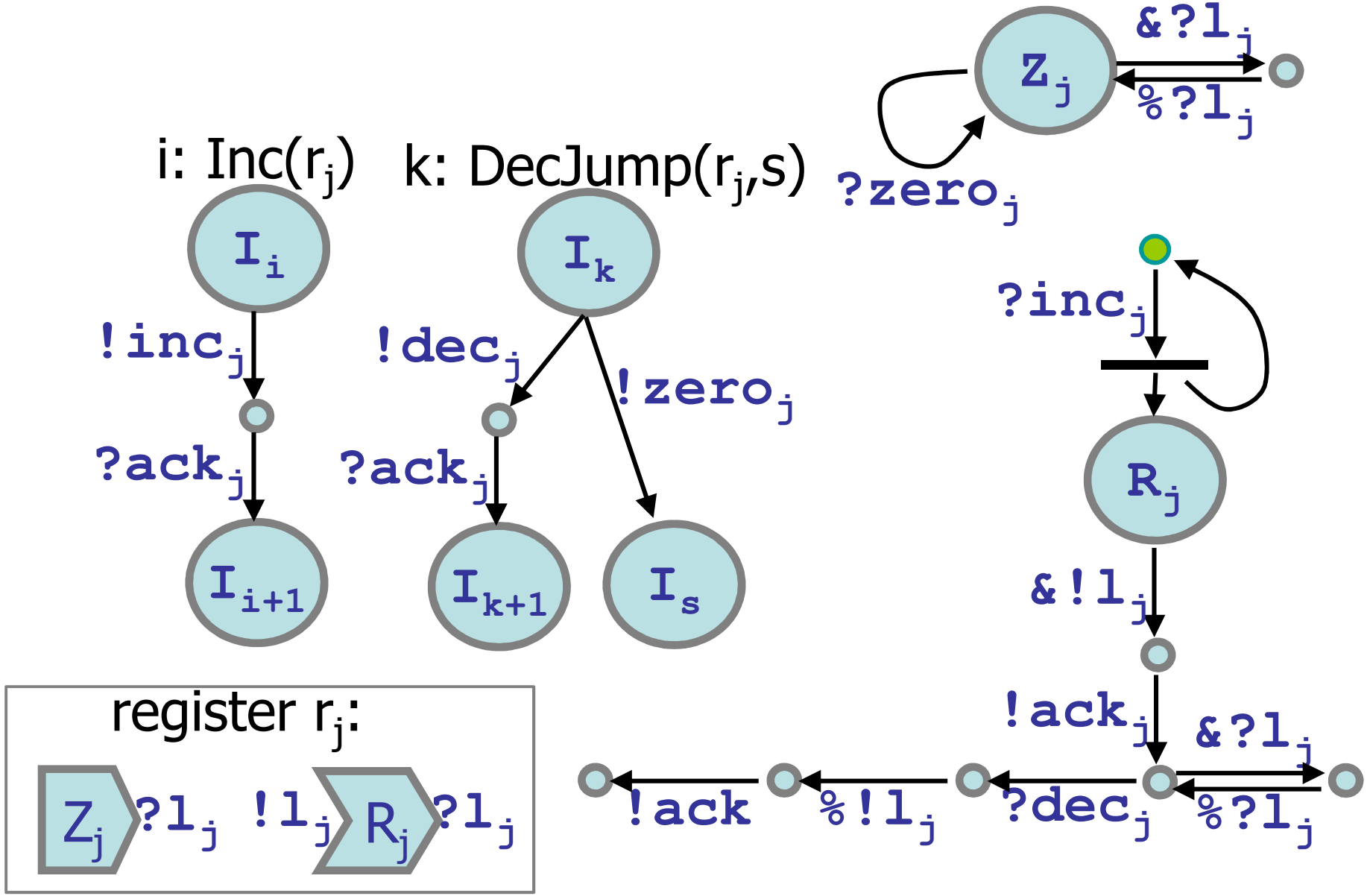
Biochemistry = Interaction + Complexation

- Complexation is what proteins "do", in contrast to simpler chemicals.



- Leading to a process algebra that we call the Biochemical Ground Form (BGF).

RAM encoding in BGF



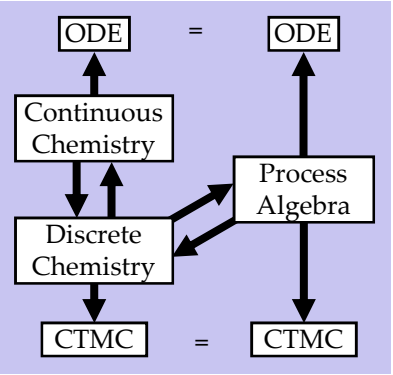
Expressiveness of Biochemistry

- Basic chemistry (FSRN, or CGF) is not Turing-complete
- Biochemistry (FSRN + complexation, or BGF) is Turing-complete.
- More powerful process algebras of course *are* Turing complete
  - They (e.g. π-calculus) include BGF, but they also have mechanisms that are not directly biologically justifiable.
  - In BGF we have in a sense the minimal biologically-inspired extension of FSRN, and it is already Turing-complete.
- Intrinsic to biochemistry (but not to simple chemistry) is a Turing-complete mechanism.

Conclusions

Conclusions

- Connections between modeling approaches
  - Connecting the discrete/concurrent/stochastic/molecular approach
  - to the continuous/sequential/deterministic/population approach



- Connecting syntax with semantics
  - Syntax = model presentation (equations/programs/diagrams/blobs etc.)
  - Semantics = state space (generated by the syntax)

- Ultimately, connections between analysis techniques
  - We need (and sometimes have) good semantic techniques to analyze state spaces (e.g. calculus, but also increasingly modelchecking)
  - But we need equally good syntactic techniques to structure complex models (e.g. compositionality) and analyze them (e.g. process algebra)