Biochemical Systems as Reactive Systems

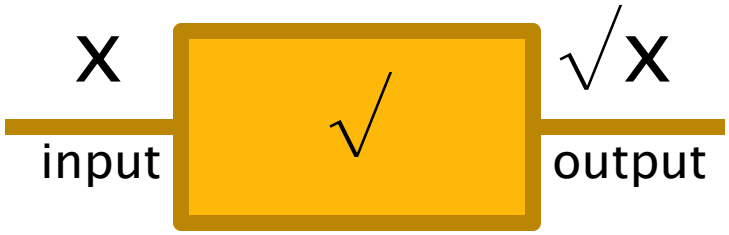
Luca Cardelli
Microsoft Research

Cambridge 2012-02-20
http://lucacardelli.name

Processes and Functions

Functions

$$f(x) = \sqrt{x}$$



"read input into x; then write $\sqrt{x}$ to output"

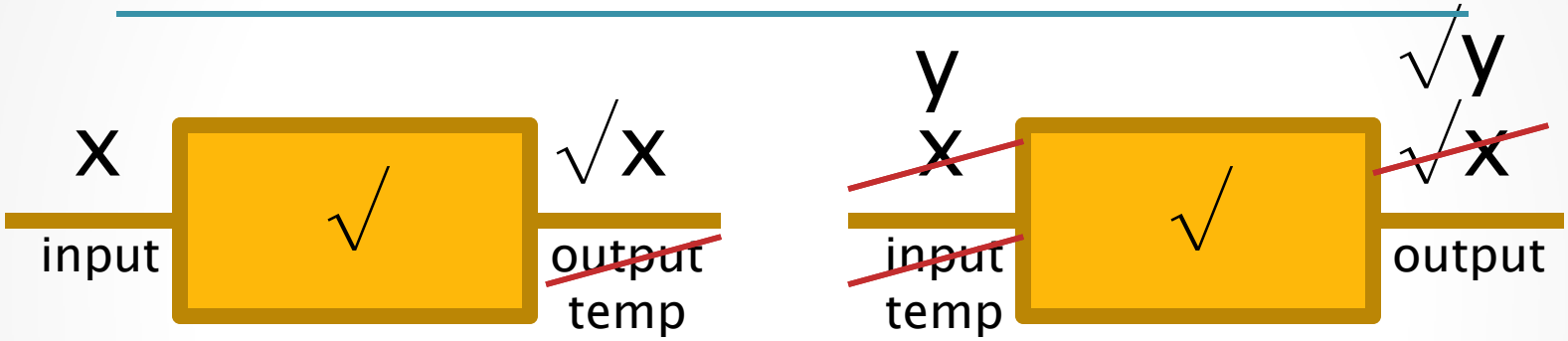
function *read* *write* *channels*

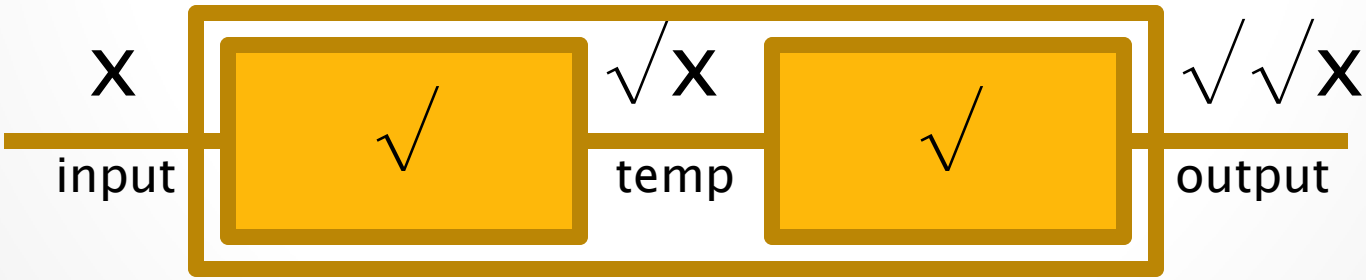
$$f = ?input(x);\ !output(\sqrt{x})$$

(binding) input variable *output expression* *(bound) variable occurrence*

Composing Functions

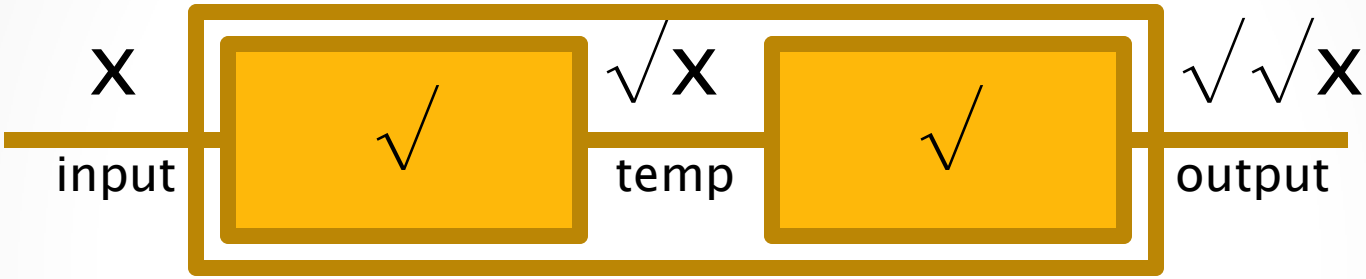
$$g(x) = (f \circ f)(x) \quad (= f(f(x)))$$





Composing Functions

$$g(x) = (f \circ f)(x)$$



"create a *new* channel and use it to compose two copies of f"

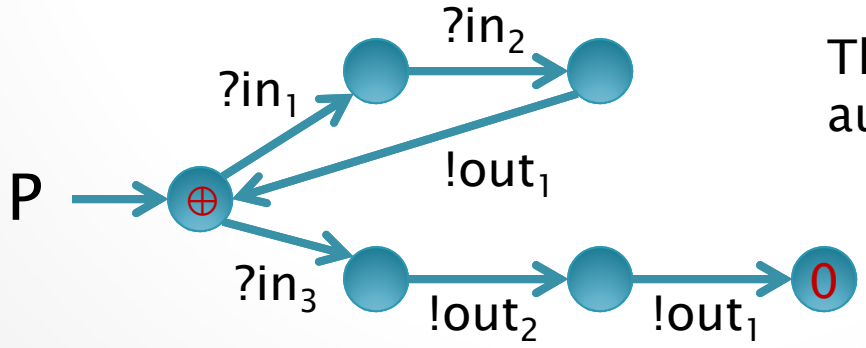
channel creation (restriction/hiding/boxing)

(parallel/process) composition

$$g = (\nu\ temp)$$
$$?input(x);\ !temp(\sqrt{x})\ |$$
$$?temp(y);\ !output(\sqrt{y})$$

Many inputs and outputs

$$P = ?in_1(x);\ ?in_2(y);\ !out_1(x+y);\ P$$
$$\oplus\ ?in_3(z);\ !out_2(\sqrt{z});\ !out_1(2z);\ 0$$



The 'skeleton' automaton

That's π-calculus

- To compose processes P we need:
  - Composition: P | P (with identity elem. 0)
  - Channel cration: (ν x) P (with x bound in P)
  - Recursion: *P (equal to P | *P)
- To execute actions we need:
  - Channel reading: ?c(x); P (with x bound in P)
  - Channel writing: !c(M); P (with message M)
  - Choice: P ⊕ P (with identity elem. 0)
- … and channels can be sent as messages!

Generalizing Functions and Automata

- Unlike functions…
  - Processes have multiple, explicitly named, input and output channels.
  - Processes can run in *parallel* , can *deadlock* on their inputs, and can be *nondeterministic* in their outputs.

- Unlike automata (FSA)…
  - Processes can transmit data (not just change state).
  - While automata ‘talk’ to input strings, processes ‘talk’ to other processes: processes are communicating automata.
  - Processes are not “finite state”; they can express unbounded computation in time (divergence) and space (proliferation).

Algebraic Properties

- Functions have one binder and one rule:
  - Function application:

    If $f(x) =_{def} M\{x\}$ then $f(a) = M\{a/x\}$

- Processes have two binders and two rules:
  - Communication (input '?' binder)

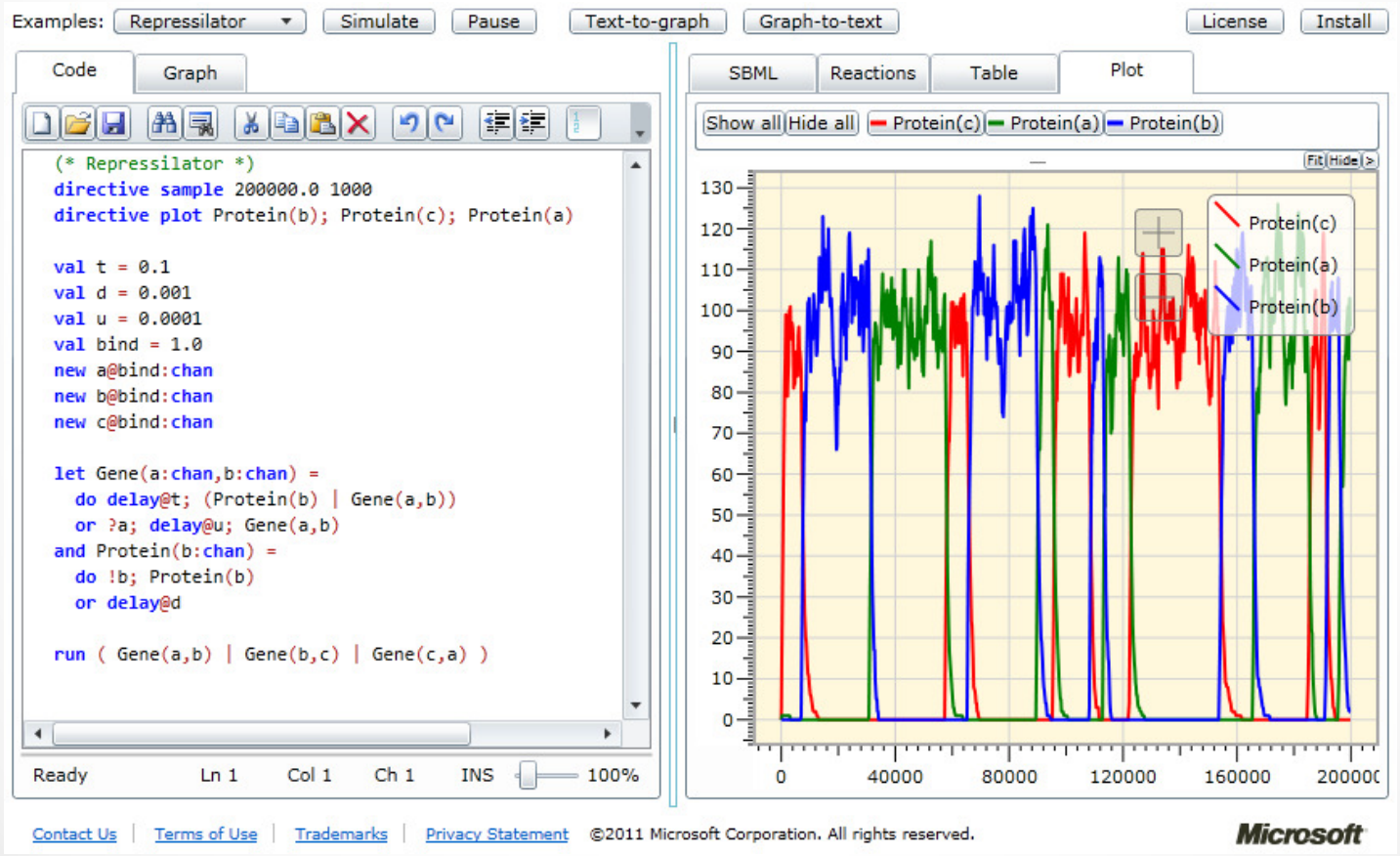
    $(?c(x);P\{x\}) \oplus P' \mid (!c(a);Q) \oplus Q' = P\{a/x\} \mid Q$

  - Scope extrusion (new 'ν' binder)

    If x not occurring in Q then $((\nu\ x)P) \mid Q = (\nu\ x)(P|Q)$

Implementations

- SPiM (Stochastic Pi Machine)
  - http://lepton.research.microsoft.com/VisualSPiM/
  - Runs in a browser with Silverlight.



Processes and Chemistry

Continuous Chemical Systems

Reactions:

A	$\rightarrow^r$	$B_1 + \ldots + B_n$	Degradation
$A_1 + A_2$	$\rightarrow^r$	$B_1 + \ldots + B_n$	Asymmetric Collision
$A + A$	$\rightarrow^r$	$B_1 + \ldots + B_n$	Symmetric Collision

Continuous reaction kinetics, respectively:

$[A]^\bullet = -r[A]$	Exponential Decay
$[A_i]^\bullet = -r[A_1][A_2]$	Mass Action Law
$[A]^\bullet = -2r[A]^2$	Mass Action Law

(assuming $A \neq B_i \neq A_j$ for all i,j)

π-calculus for Chemistry

- To compose *soups* P we need:
  - Stochastic channels: $(\nu\ x_r)\ P$ — r is the rate of an exponential distribution: the rate of communication on that channel
  - Composition: $P \mid P$ (with identity elem. 0)
  - Recursion: $*P$ (equal to $P \mid *P$)

- To execute *species* we need:
  - Collision: $?x_r;\ P$ (with no input variables)
  - Co-collision: $!x_r;\ P$ (with no output messages)
  - Delay: $\tau_r;\ P$ ($= (\nu\ x_r)\ ?x_r;P|!x_r;0$ for any x not in P)
  - Choice: $P \oplus P$ (with identity elem. 0)

Discrete Chemical Systems (1)

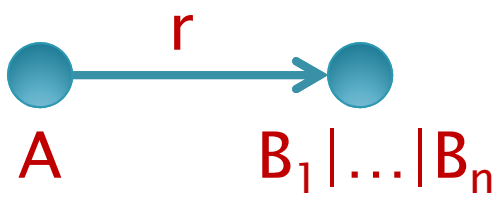
Reaction:

$A \rightarrow^{r} B_1 + \ldots + B_n$

Discrete reaction kinetics:

$A = \tau_r; (B_1 | \ldots | B_n)$

The mathematical meaning of that is a Continuous Time Markov Chain (for a specific set of initial conditions, e.g. a single A molecule), here represented as a transition graph:



Hence the π-calculus description abstracts from initial conditions (like ODEs). For each set of initial conditions, a CTMC can be systematically extracted from the stochastic π-calculus models.

Discrete Chemical Systems (2)

(Uniquely named) reaction:

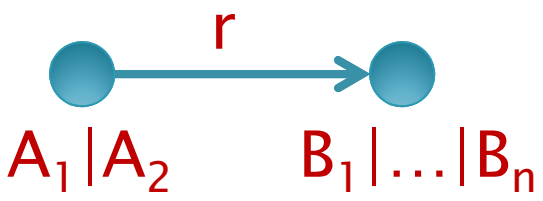
$c: \; A_1 + A_2 \;\rightarrow^{r}\; B_1 + \ldots + B_n$

Discrete reaction kinetics:

$A_1 = ?c_r; (B_1|\ldots|B_i)$ (the name of the reaction becomes the channel)

$A_2 = !c_r; (B_i|\ldots|B_n)$ (splitting results is arbitrary: $1 \leq i \leq n$)

With initial conditions $A_1|A_2$ (single molecules), the CTMC is:



Discrete Chemical Systems (3)

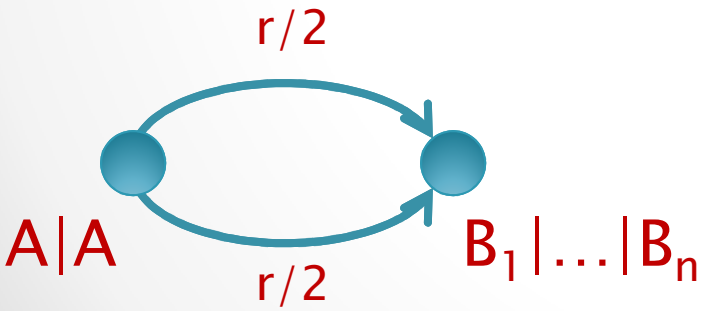
(Uniquely named) reaction:

$c: \; A + A \;\rightarrow^{r}\; B_1 + \ldots + B_n$

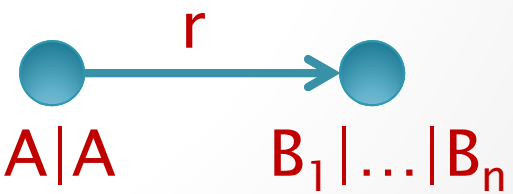
Discrete reaction kinetics:

$$A = ?c_{r/2};\ (B_1|\ldots|B_i) \oplus !c_{r/2};\ (B_i|\ldots|B_n) \qquad 1 \le i \le n$$

With initial conditions $A|A$ (two molecules), the CTMC is as follows; note that each copy of A can do an input or an output, so there are two possible paths to the outcome:



That is:



From Reactions to Processes

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

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

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

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

Interaction Matrix

channels (1 per reaction)

Half-rate for symmetric reactions

processes (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

Fill the matrix by columns:

Degradation reaction v_i: $X \rightarrow^{ki} P_i$
add $\tau;P_i$ to $\langle X,v_i \rangle$.

Asymmetric reaction v_i: $X+Y \rightarrow^{ki} P_i$
add $?;P_i$ to $\langle X,v_i \rangle$ and $!;0$ to $\langle Y,v_i \rangle$

Symmetric reaction v_i: $X+X \rightarrow^{ki} P_i$
add $?;P_i$ and $!;0$ to $\langle X,v_i \rangle$

Read out the processes by rows:

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

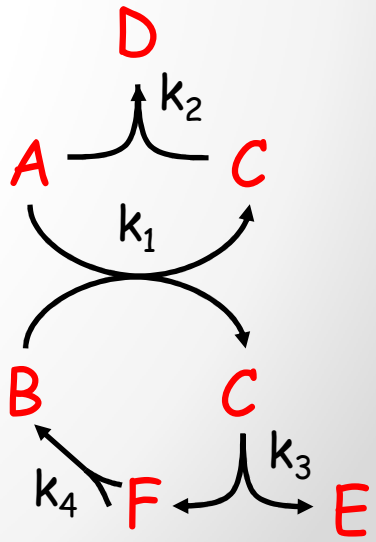
$B = !v_{1(k1)};0$

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

$D = 0$

$E = 0$

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



That Chemical System in SPiM

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

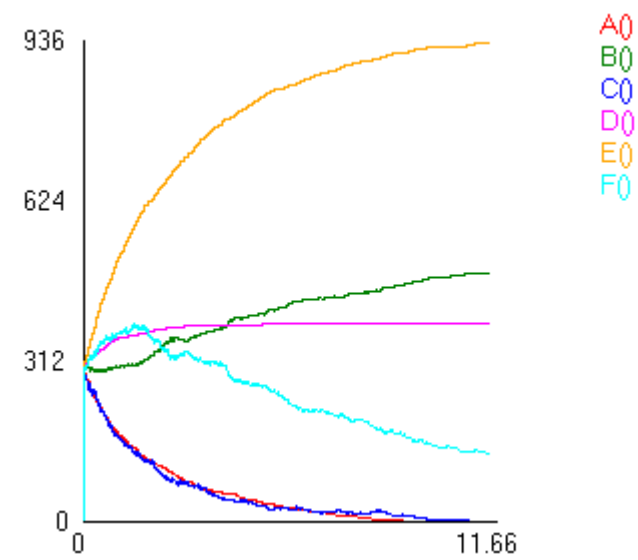
$B = !v_{1(k1)};0$

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

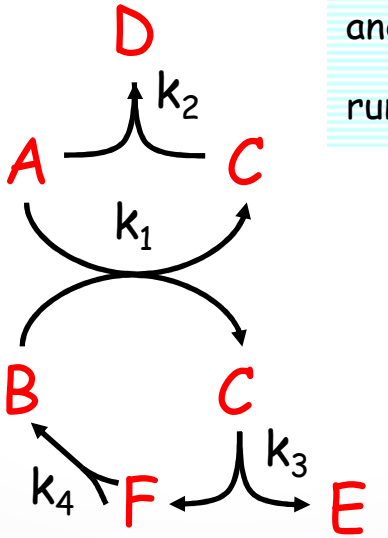
$D = 0$

$E = 0$

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



Gillespie-style stochastic simulation




```
directive sample 10.0
directive plot A(); B(); C(); D(); E(); F()

val k1 = 0.001  new v1@k1:chan
val k2 = 0.001  new v2@k1:chan
val k3 = 1.0
val k4 = 0.001  new v4@k4/2.0:chan

let A() = do ?v1;(C()|C()) or ?v2;D()
and B() = !v1
and C() = do !v2 or delay@k3;(E()|F())
and D() = ()
and E() = ()
and F() = do ?v4;B() or !v4

run 300 of (A()|B()|C()|D()|E()|F())
```

Modeling Techinques

- That is a *systematic* way to translate reactions to processes.
- But there can be *better* ways to do it.
- That is, ways that produce *more compact and/or modular models*, but with the *same kinetics*.

Ex: Catalysis

- Two reactions, same catalyst C
  - According to the general scheme the catalyst uses one channel for each reaction it catalyzes

a: $A + C \to^{r} C + B$ $\quad C = !a_r; C \oplus !b_r; C$

b: $D + C \to^{r} C + E$ $\quad A = ?a_r; B$

$D = ?b_r; E$

  - Modularizing: the catalyst has its own catalysis channel c, used for all the reactions it catalyzes:

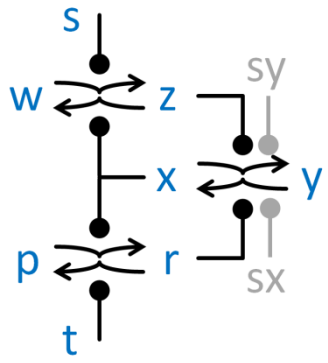
$A + C \to^{r} C + B$ $\quad C = !c_r; C$

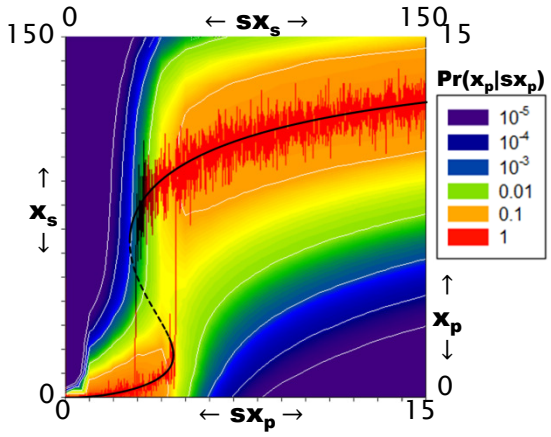
$D + C \to^{r} C + E$ $\quad A = ?c_r; B$

$D = ?c_r; E$

Ex: System Analysis

- 16-reaction abstract model of cell-cycle switch.





sx (hor axis): input value
x (ver axis): output value at equilibrium

Black line: deterministic bifurcation diagram
Red line: stochastic simulation (sx_s,x_s) at size x_{max}=150 [by SPiM]
Heatmap: discrete probability distribution (sx_p,x_p) at size x_{max}=15 [by PRISM]

(Joint work with Attila Csikasz-Nagy)

Processes and Biochemistry

π-calculus for Biochemistry

- Biochemistry here means: direct modeling of complexation and polymerization.
- We now go back to the full (and stochastic) π-calculus: we need to pass channels as messages!

Complexation

$$A + B \quad {}^{s}\!\leftrightarrow^{r} \quad A{:}B$$

There is no good notation for this reaction in chemistry: A:B is considered as a separate species (which leads to combinatorial explosion of models).

But there is a way to write this precisely in π-calculus. There is a single public *association* channel a_r at rate r, and many private *dissociations* channels d_s at rate s, one for each complexation event (created by ν):

$$\begin{aligned} A_{free} &= (\nu\ d_s)\ !a_r(d_s);\ A_{bound}(d_s) \\ A_{bound}(d_s) &= !d_s;\ A_{free} \end{aligned}$$

$$\begin{aligned} B_{free} &= ?a_r(d_s);\ B_{bound}(d_s) \\ B_{bound}(d_s) &= ?d_s;\ B_{free} \end{aligned}$$

Note that we are describing A *independently* of B: as in the catalysis example, A could form complexes with many different species over the a_r channel.

More compactly:

$A = (\nu\ d_s)\ !a_r(d_s);\ !d_s;\ A$
$B = ?a_r(d_s);\ ?d_s;\ B$

Polymerization

- Polymerization is iterated complexation
  - It can be represente in π-calculus *finitely*, with one process (definition) for each monomer.
  - Note that polymerization cannot be described *finitely* in chemistry (or ODEs) because there it needs one reaction for each *length* of polymer.
  - The reason it works in π-calculus is because of the ν operator. It enables the finite representation of systems of potentially unbounded complexity.
  - Like real biochemistry, where the structure of each monomer is coded in a finite piece of DNA, and yet unbounded-length polymers happen.

Conclusions

Conclusions

- π-calculus
  - A mathematical notation for reactive systems
  - In stochastic form, suitable for representing discrete chemistry, biochemistry, etc.
  - Some unique properties: ability to finitely express systems of unbounded complexity, like networks of complexing proteins.

- Further Reading
  - R. Milner: **Communicating and Mobile Systems: The Pi Calculus**
  - A. Regev, E. Shapiro. **Cellular Abstractions: Cells as Computation.** NATURE vol 419, 2002-09-26, 343.
  - L. Cardelli: **From Processes to ODEs by Chemistry.** TCS 2008, DOI: http://dx.doi.org/10.1007/978-0-387-09680-3_18
  - A. Phillips,L. Cardelli, **A Correct Abstract Machine for the Stochastic Pi-calculus**, in Concurrent Models in Molecular Biology, 2004.