Modeling Bio-bricks Using Recent Extensions to Kappa

Elaine Murphy

October 26, 2009

School of informatics

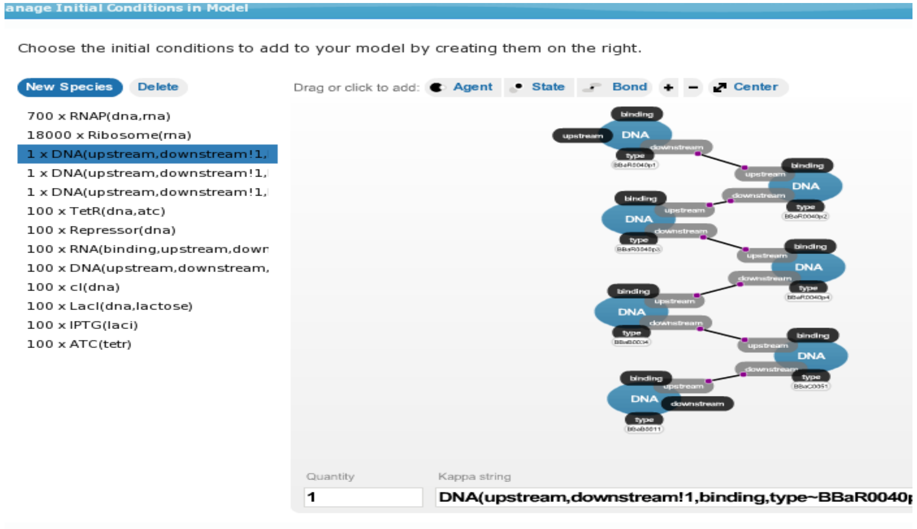
Previous Lecture

Ty's rule-based modelling of Bio-bricks:

- encapsulates mechanisms of transcription/translation clearly and concisely;
- represents mechanisms at the level of Bio-bricks;
- includes good examples that show how the models works,
  i.e Elowitz repressilator, Toggle switch.

Disadvantages of Modelling Approach

- Readability -
  The initial conditions and the rules are quite difficult to read. It is not obvious what the rules are doing unless you are very familiar with the rule set.



anage Initial Conditions in Model

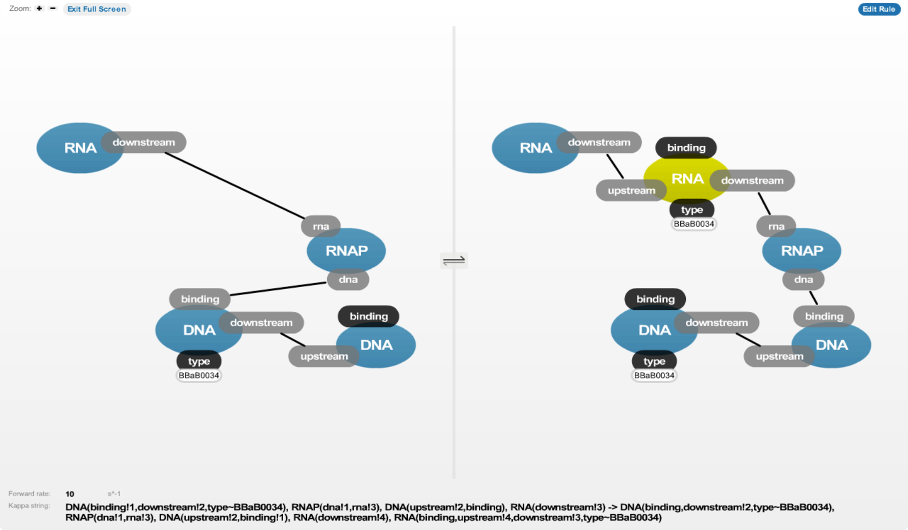
Choose the initial conditions to add to your model by creating them on the right.

700 x RNAP(dna,rna)
18000 x Ribosome(rna)
1 x DNA(upstream,downstream!1,
1 x DNA(upstream,downstream!1,
1 x DNA(upstream,downstream!1,
100 x TetR(dna,atc)
100 x Repressor(dna)
100 x RNA(binding,upstream,dowr
100 x DNA(upstream,downstream,
100 x cI(dna)
100 x LacI(dna,lactose)
100 x IPTG(laci)
100 x ATC(tetr)

Quantity: 1

Kappa string: DNA(upstream,downstream!1,binding,type~BBaR0040p

- Modularity -
  Any rule that mentions a specific type of DNA is specific only for that piece.



Forward rate: 10 s^-1

Kappa string: DNA(binding!1,downstream!2,type~BBaB0034), RNAP(dna!1,rna!3), DNA(upstream!2,binding), RNA(downstream!3) -> DNA(binding,downstream!2,type~BBaB0034), RNAP(dna!1,rna!3), DNA(upstream!2,binding!1), RNA(downstream!4), RNA(binding,upstream!4,downstream!3,type~BBaB0034)

- Out of 17 basic rules in the rule set, 14 of them are type specific.
- This means that each of these 14 rules have to be duplicated for each new piece of DNA being entered into a model.

Aim of Using Extensions to model Bio-bricks

- To address these issues of readability and modularity using recent extensions to Kappa.
- To keep the representation similar to Bio-brick representation to make modelling easier.
- To keep the concise and detailed representation of Ty's work.

Bio-Bricks Parts and Information

There are 4 main types of parts in the Bio-brick registry.
Promoters, RBS, PCS, Terminators.
They (should) have the following information parameters.

- Promoter -
  Binding affinities for repressors/activators,
  Rate of recruitment of RNAP
- RBS -
  Rate of transcription
  Rate of recruitment of Ribosomes
- PCS -
  Rate of transcription

Rate of translation
Rate of degradation of Protein

- Terminator -
  Percentage of successful termination

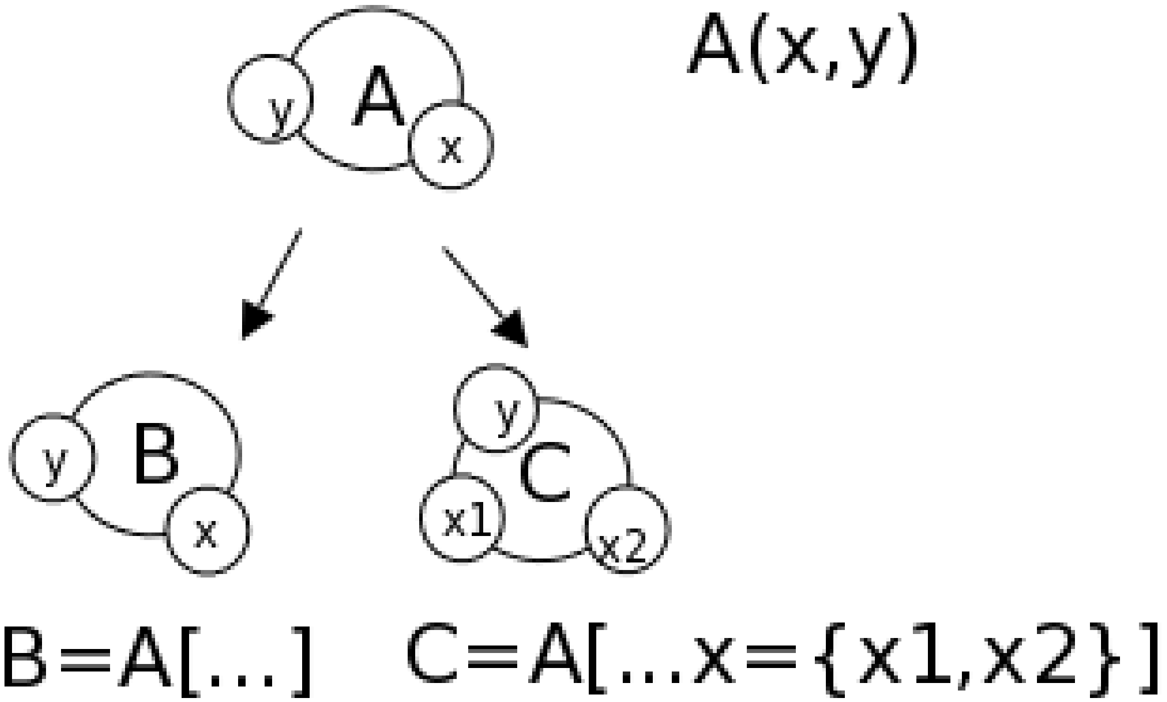
Hierarchies in Kappa

In normal Kappa there is no sense of classes or types of agents. This extension of Kappa addresses this by introducing agent hierarchies.

- Agents can be declared in the normal manner with a name and a set of sites, eg $A(x, y)$
  OR

- Agents can be declared as children of existing agents.
  - Children agents inherit sites from their parents.
  - Children can also have additional or replicated sites.

Example of Hierarchy



A(x,y)

B=A[...]

C=A[...x={x1,x2}]

Rules with Hierarchies

- Rules can be written using parent agents or child agents or a mixture of both types of agents.

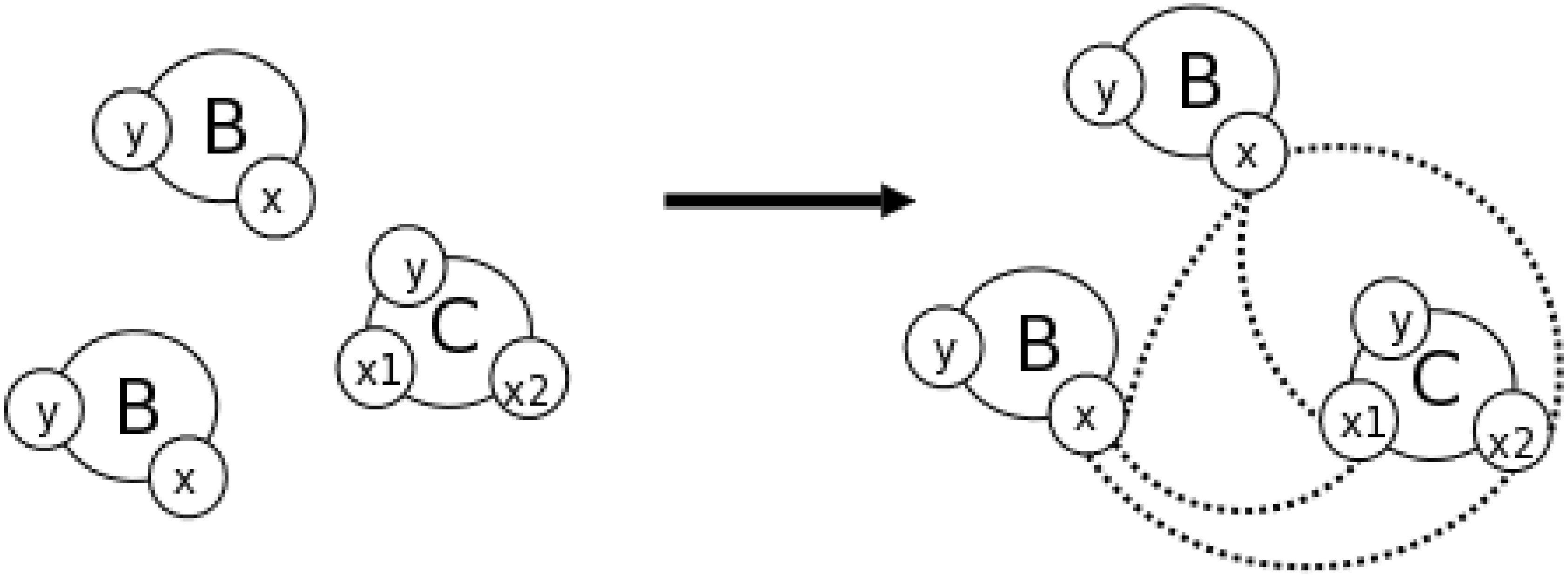
- Rules containing parents agents are called generic rules.

- Any rule that can be applied to a parent agent can be applied to its children.

- If a rule mentions a site that is duplicated in a child, the rule can be applied using either of the duplicated sites.

- The initial conditions of a model can not contain parent agents.

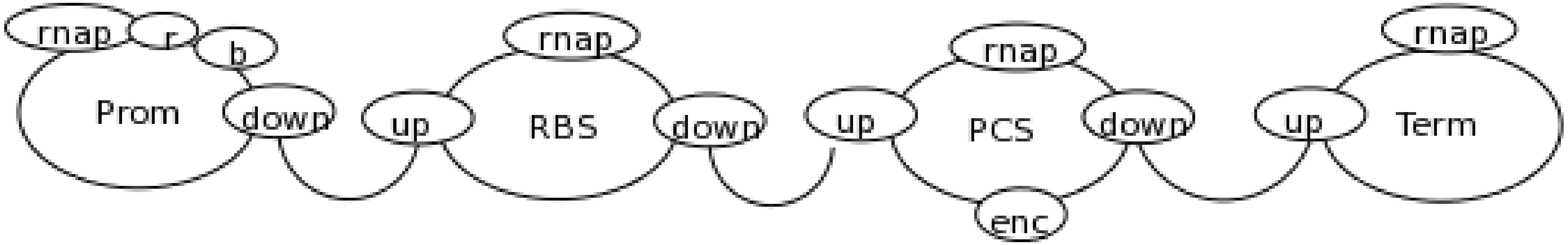
Rule: A(x),A(x) -> A(x!1),A(x!1)



Hierarchies in Bio-brick Modelling

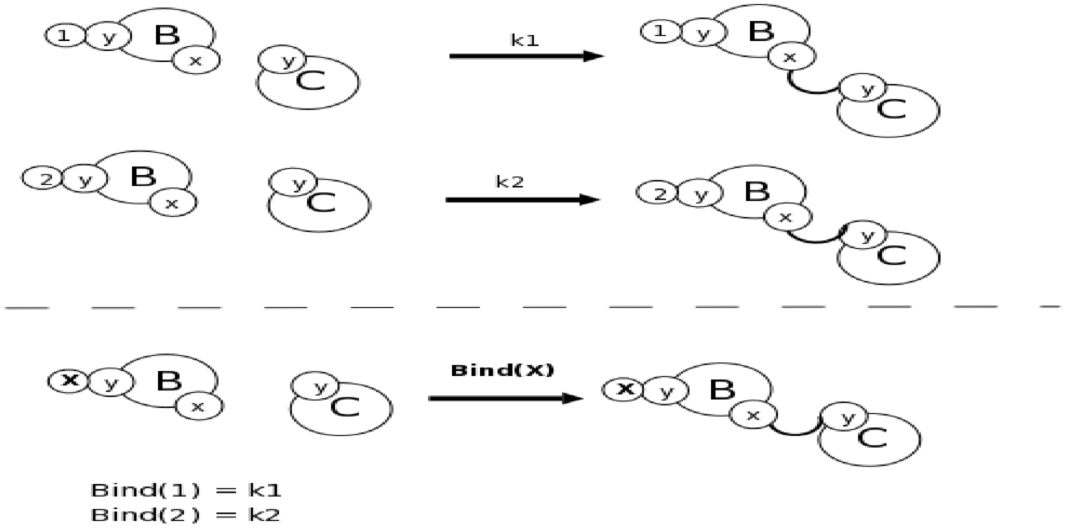
Agents:

$$
\begin{array}{cc}
DNA(up, down, rnap, name) & Term = DNA[...] \\
mRNA(up, down, ribo, enc) & PCS = DNA[..., enc] \\
RNAP(dna, rna) & RBS = DNA[...] \\
Ribosome(rna) & Prom = DNA[..., r, b] \\
Protein(name) &
\end{array}
$$



Coloured Kappa

Coloured Kappa is an extension of Kappa that allows variables in rules and allows the rate of a rule to be a function of the variables.



In Bio-brick modelling we are only going to use colouring of internal states.

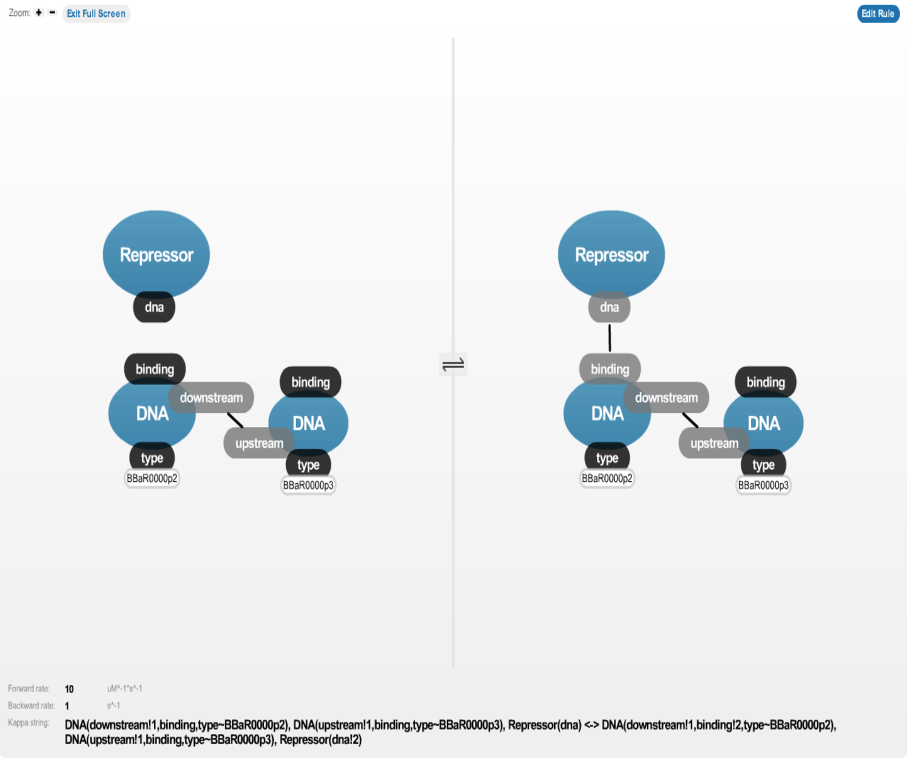
Modelling of Bio-brick Using Extensions

- The rule set will be an isomorphism of Ty's basic rule set, i.e each rule in Ty's basic rule set will match to one rule in the new rule set.

- However for the model specific rule set it will be a many-to-one mapping.

- For example Ty's model of the Elowitz repressilator has 49 rules but to model it in this manner just needs 17 rules.

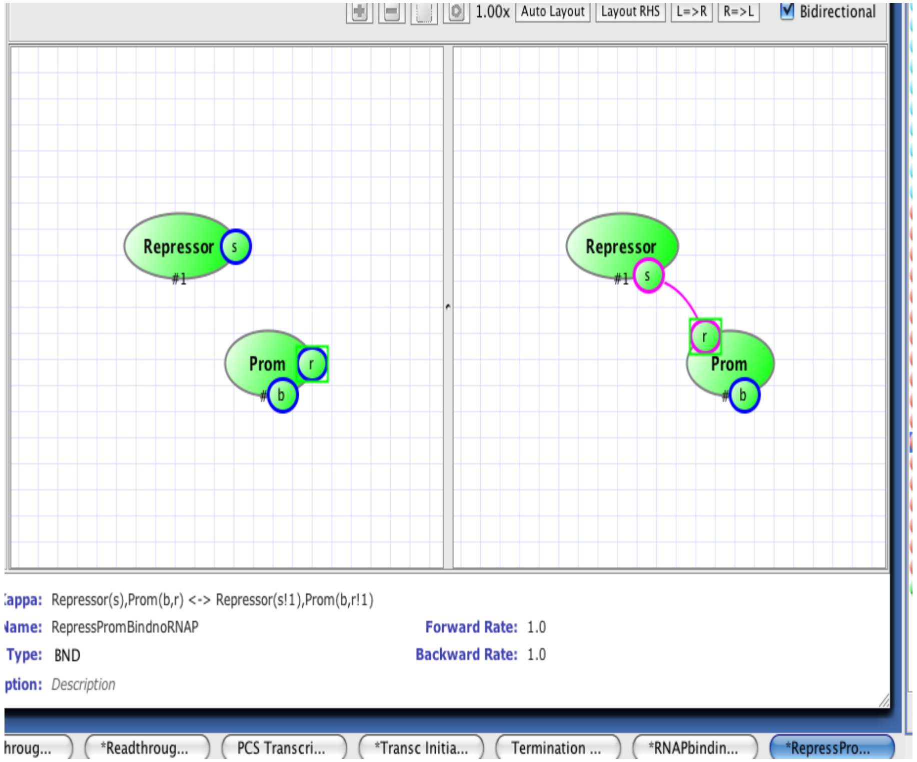
Repressor Binding to Promoter without RNAP



Forward rate: 10 uM^-1*s^-1

Backward rate: 1 s^-1

Kappa string: DNA(downstream!1,binding,type~BBaR0000p2), DNA(upstream!1,binding,type~BBaR0000p3), Repressor(dna) <-> DNA(downstream!1,binding!2,type~BBaR0000p2), DNA(upstream!1,binding,type~BBaR0000p3), Repressor(dna!2)



☑ Bidirectional

appa: Repressor(s),Prom(b,r) <-> Repressor(s!1),Prom(b,r!1)

Name: RepressPromBindnoRNAP

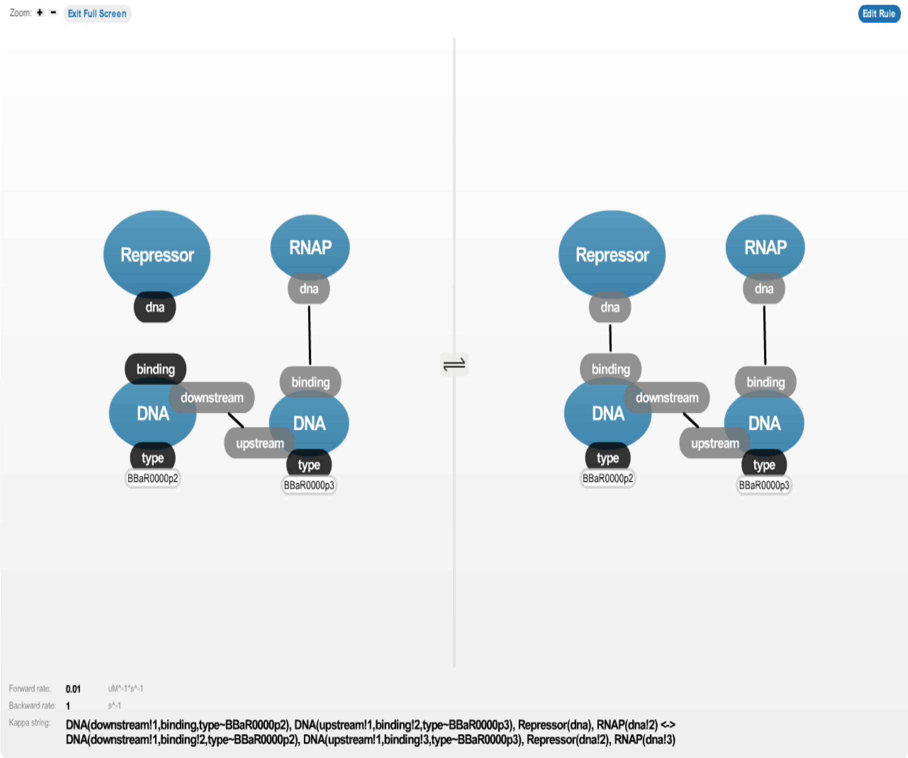
Type: BND

ption: Description

Forward Rate: 1.0

Backward Rate: 1.0

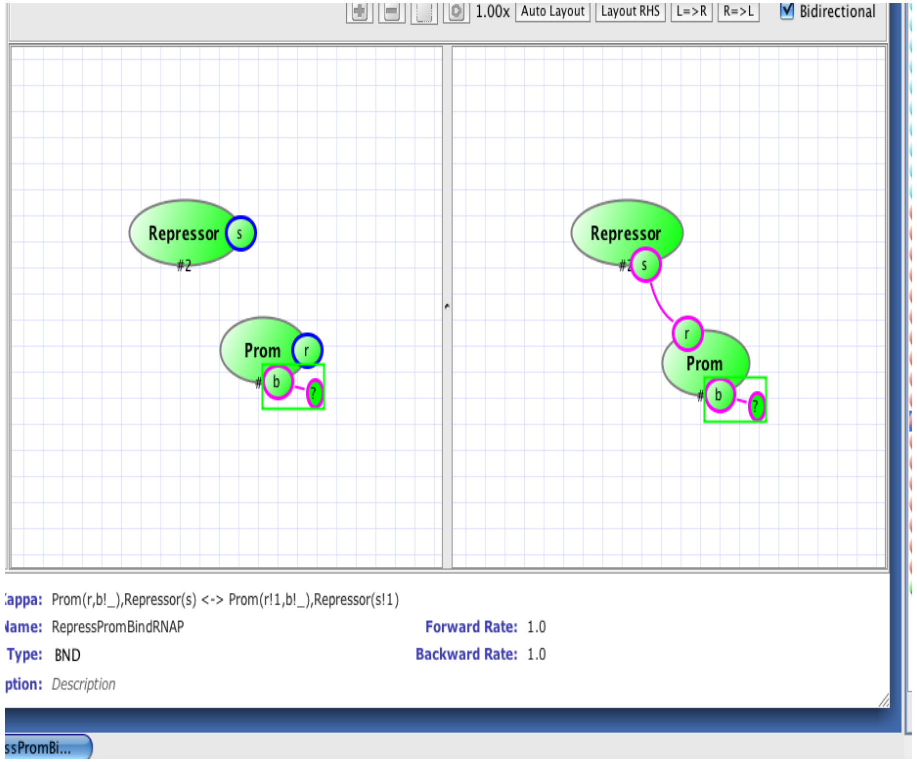
Repressor Binding to Promoter with RNAP



Forward rate: 0.01 uM^-1*s^-1

Backward rate: 1 s^-1

Kappa string: DNA(downstream!1,binding,type~BBaR0000p2), DNA(upstream!1,binding!2,type~BBaR0000p3), Repressor(dna), RNAP(dna!2) <-> DNA(downstream!1,binding!2,type~BBaR0000p2), DNA(upstream!1,binding!3,type~BBaR0000p3), Repressor(dna!2), RNAP(dna!3)



☑ Bidirectional

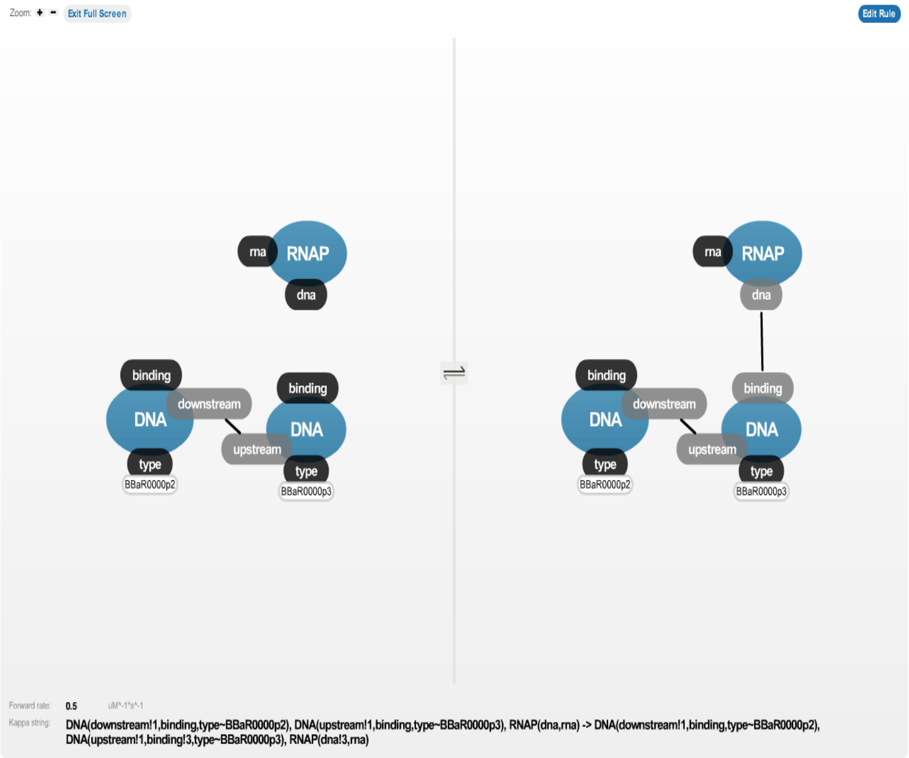
:appa: Prom(r,b!_),Repressor(s) <-> Prom(r!1,b!_),Repressor(s!1)

lame: RepressPromBindRNAP | Forward Rate: 1.0

Type: BND | Backward Rate: 1.0

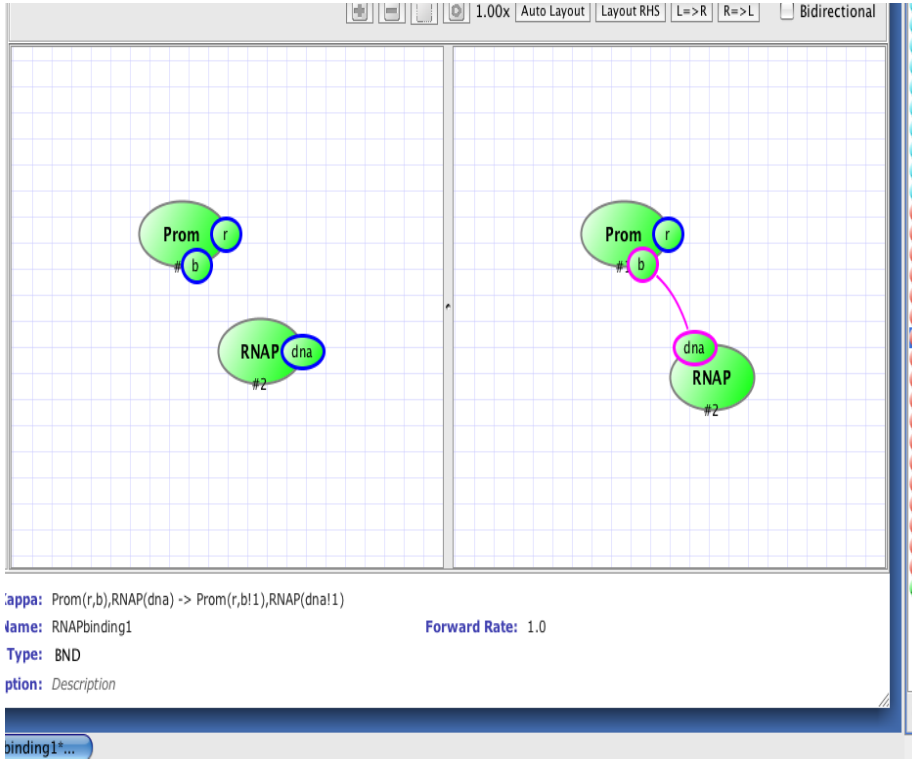
ption: *Description*

RNAP Binding to Promoter without Repressor



Forward rate: 0.5 uM^-1*s^-1

Kappa string: DNA(downstream!1,binding,type~BBaR0000p2), DNA(upstream!1,binding,type~BBaR0000p3), RNAP(dna,rna) -> DNA(downstream!1,binding,type~BBaR0000p2), DNA(upstream!1,binding!3,type~BBaR0000p3), RNAP(dna!3,rna)



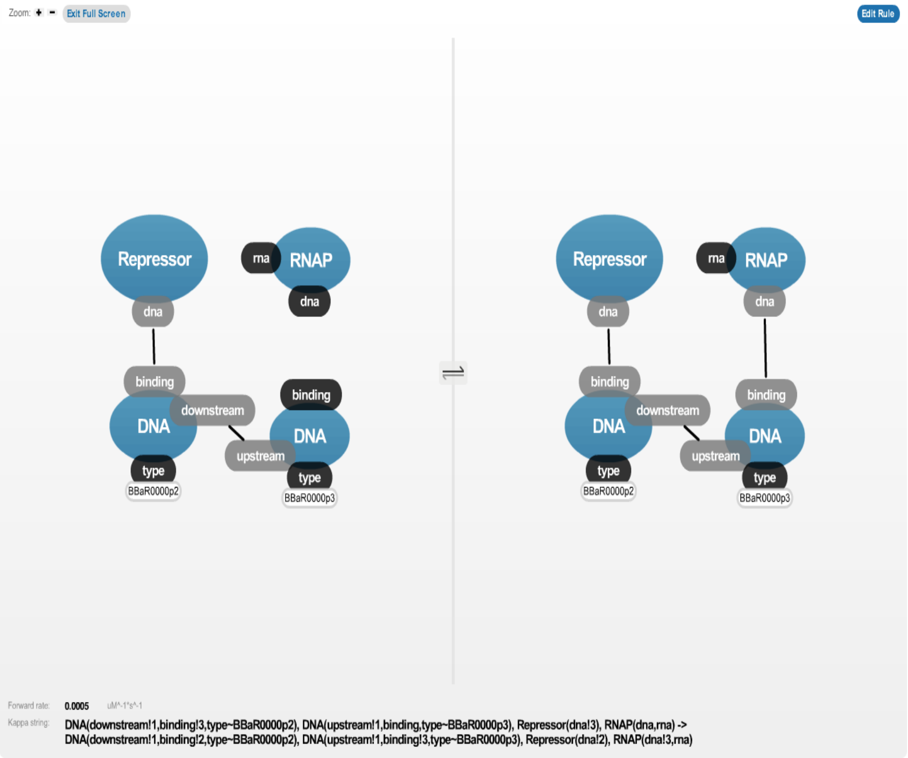
appa: Prom(r,b),RNAP(dna) -> Prom(r,b!1),RNAP(dna!1)

lame: RNAPbinding1 Forward Rate: 1.0

Type: BND

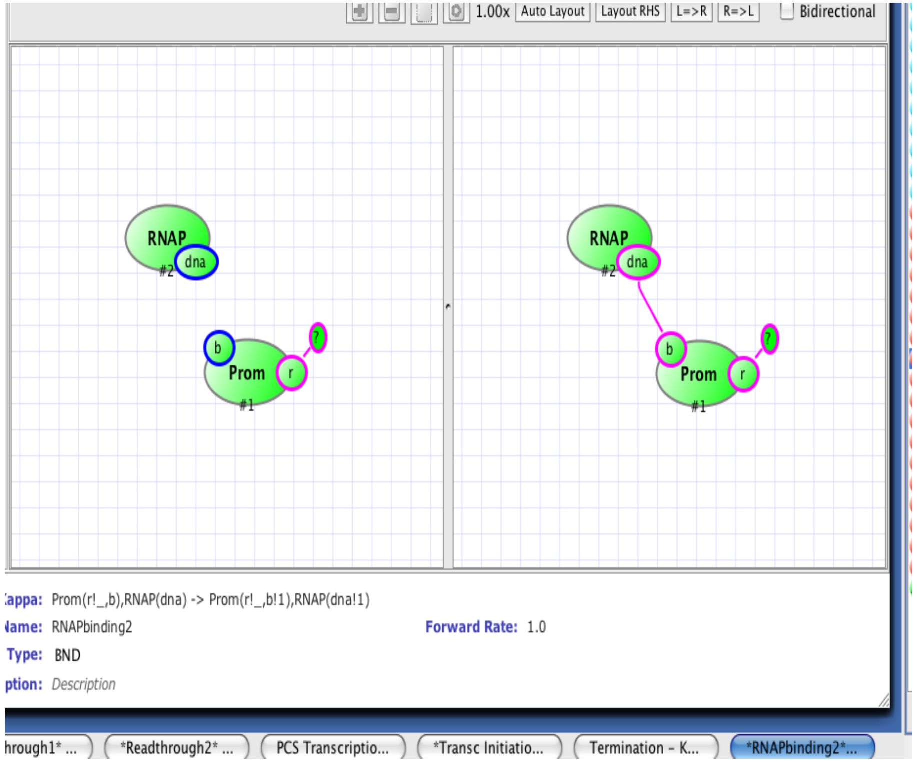
ption: *Description*

RNAP Binding to Promoter with Repressor



Forward rate: 0.0005 uM^-1*s^-1

Kappa string: DNA(downstream!1,binding!3,type~BBaR0000p2), DNA(upstream!1,binding,type~BBaR0000p3), Repressor(dna!3), RNAP(dna,rna) -> DNA(downstream!1,binding!2,type~BBaR0000p2), DNA(upstream!1,binding!3,type~BBaR0000p3), Repressor(dna!2), RNAP(dna!3,rna)



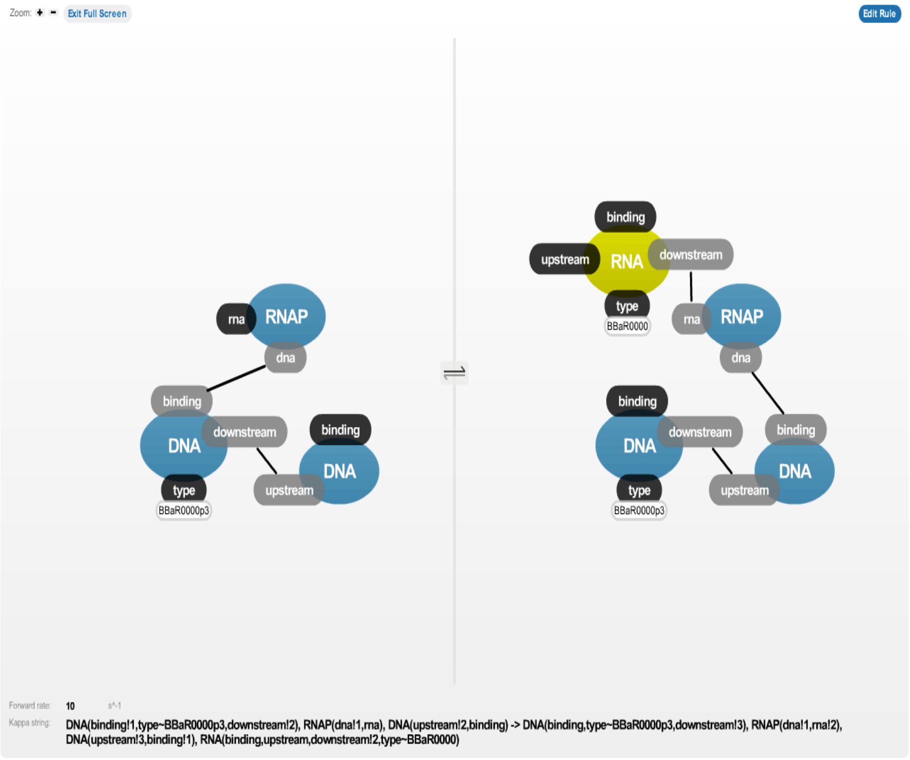
:appa: Prom(r!_,b),RNAP(dna) -> Prom(r!_,b!1),RNAP(dna!1)

lame: RNAPbinding2 Forward Rate: 1.0

Type: BND

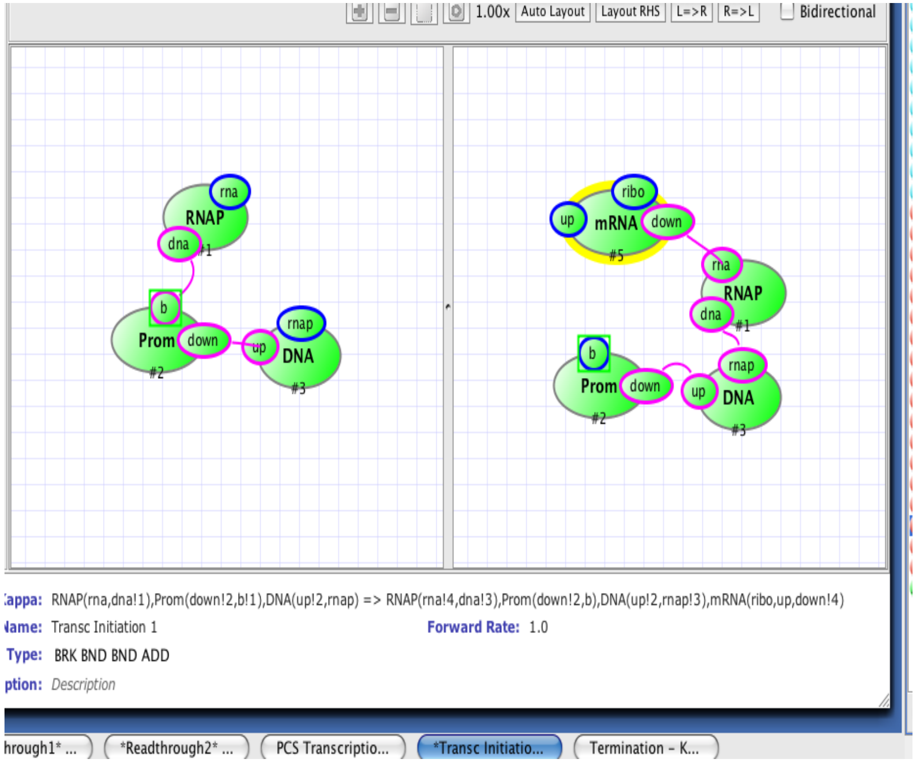
ption: *Description*

Transcription Initiation



Forward rate: 10 s^-1

Kappa string: DNA(binding!1,type~BBaR0000p3,downstream!2), RNAP(dna!1,rna), DNA(upstream!2,binding) -> DNA(binding,type~BBaR0000p3,downstream!3), RNAP(dna!1,rna!2), DNA(upstream!3,binding!1), RNA(binding,upstream,downstream!2,type~BBaR0000)



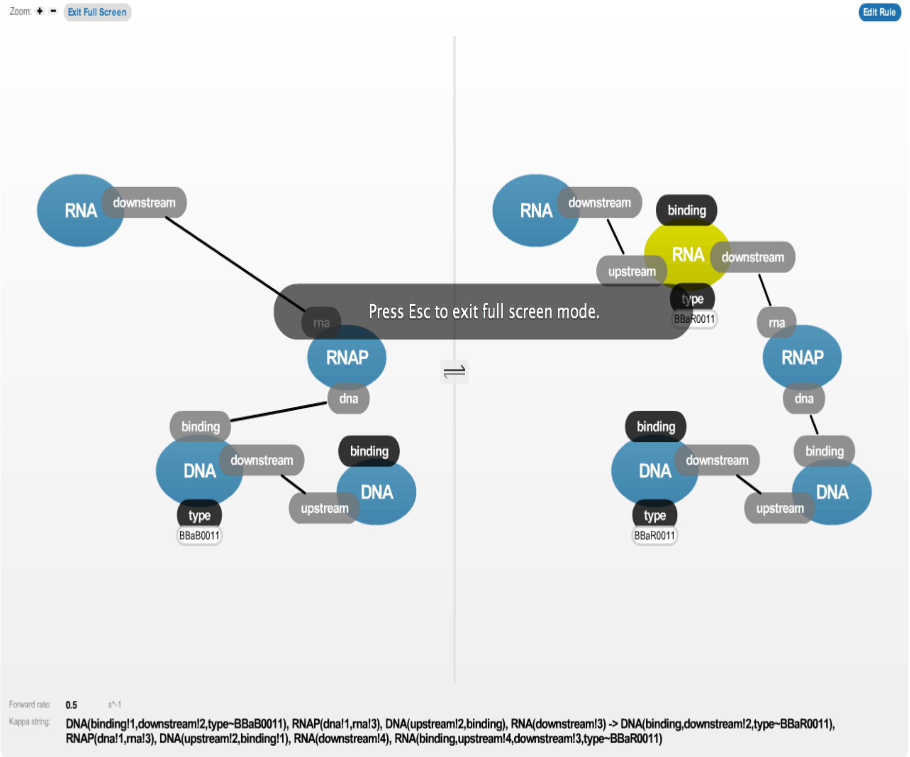
Kappa: RNAP(rna,dna!1),Prom(down!2,b!1),DNA(up!2,rnap) => RNAP(rna!4,dna!3),Prom(down!2,b),DNA(up!2,rnap!3),mRNA(ribo,up,down!4)

Name: Transc Initiation 1 Forward Rate: 1.0

Type: BRK BND BND ADD

ption: *Description*

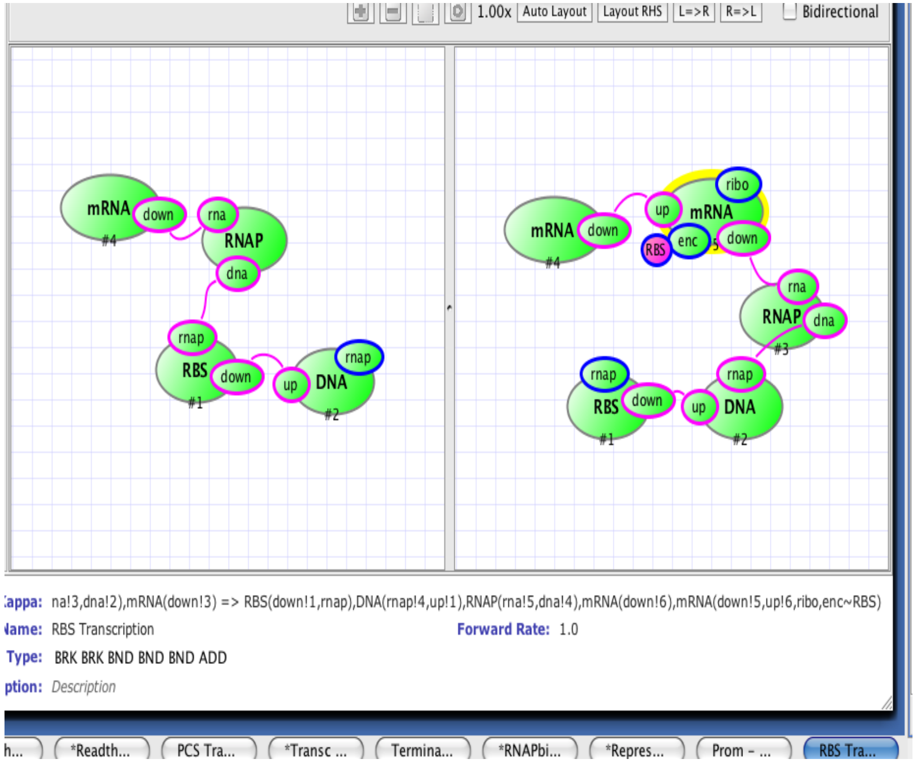
RBS Transcription



Press Esc to exit full screen mode.

Forward rate: 0.5 s^-1

Kappa string: DNA(binding!1,downstream!2,type~BBaB0011), RNAP(dna!1,rna!3), DNA(upstream!2,binding), RNA(downstream!3) -> DNA(binding,downstream!2,type~BBaR0011), RNAP(dna!1,rna!3), DNA(upstream!2,binding!1), RNA(downstream!4), RNA(binding,upstream!4,downstream!3,type~BBaR0011)



Kappa: na!3,dna!2),mRNA(down!3) => RBS(down!1,rnap),DNA(rnap!4,up!1),RNAP(rna!5,dna!4),mRNA(down!6),mRNA(down!5,up!6,ribo,enc~RBS)

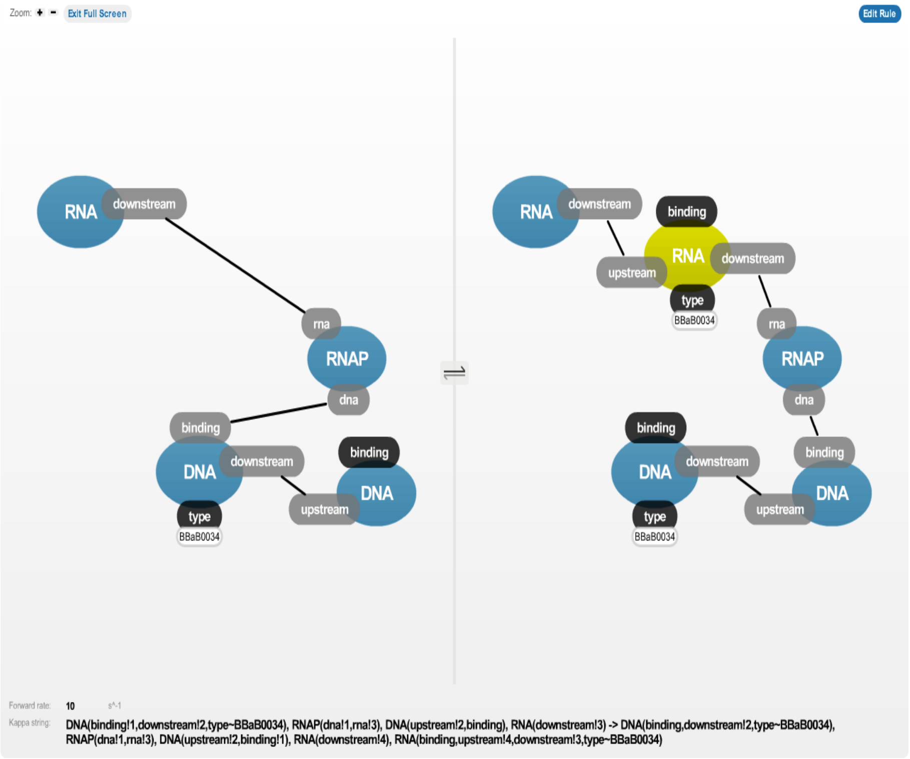
Name: RBS Transcription

Forward Rate: 1.0

Type: BRK BRK BND BND BND ADD

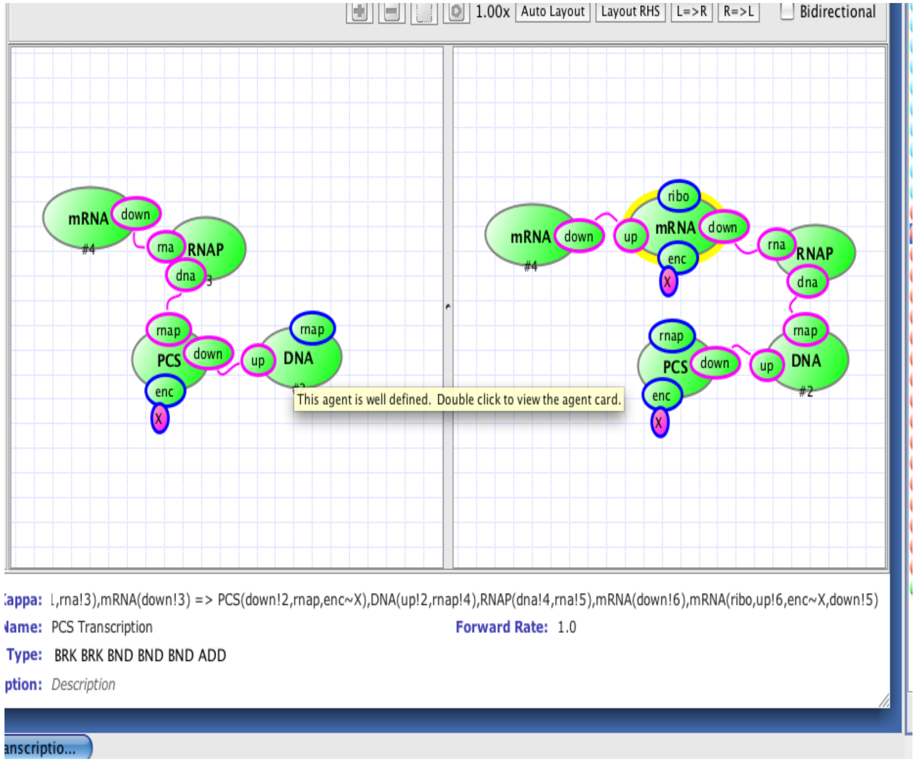
ption: *Description*

PCS Transcription



Forward rate: 10 s^-1

Kappa string: DNA(binding!1,downstream!2,type~BBaB0034), RNAP(dna!1,rna!3), DNA(upstream!2,binding), RNA(downstream!3) -> DNA(binding,downstream!2,type~BBaB0034), RNAP(dna!1,rna!3), DNA(upstream!2,binding!1), RNA(downstream!4), RNA(binding,upstream!4,downstream!3,type~BBaB0034)



Kappa: l,rna!3),mRNA(down!3) => PCS(down!2,rnap,enc~X),DNA(up!2,rnap!4),RNAP(dna!4,rna!5),mRNA(down!6),mRNA(ribo,up!6,enc~X,down!5)

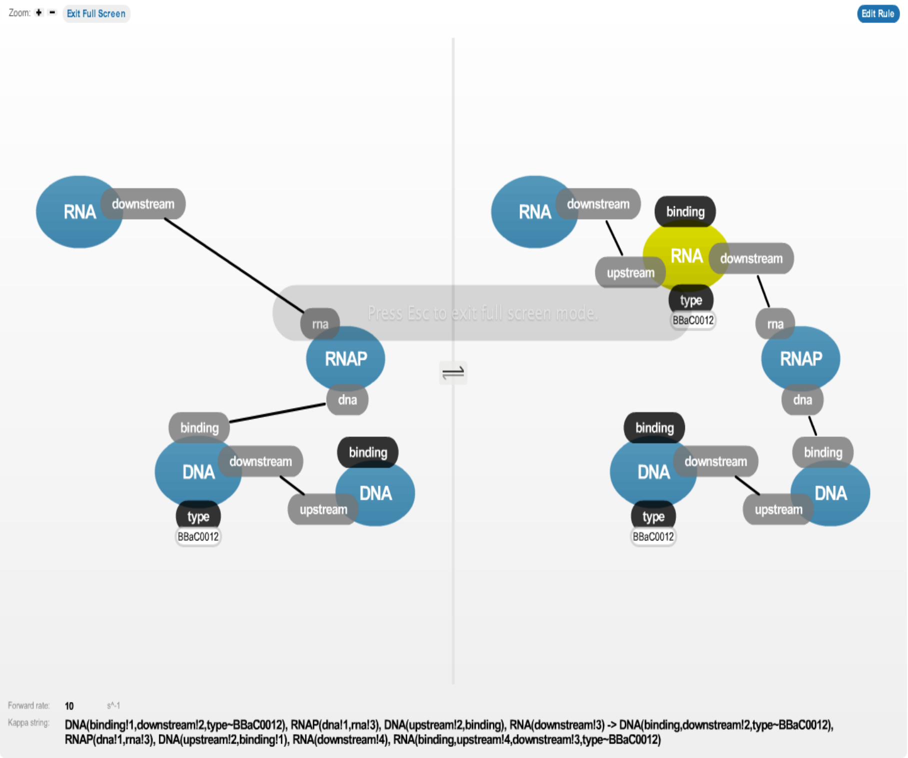
Name: PCS Transcription

Forward Rate: 1.0

Type: BRK BRK BND BND BND ADD

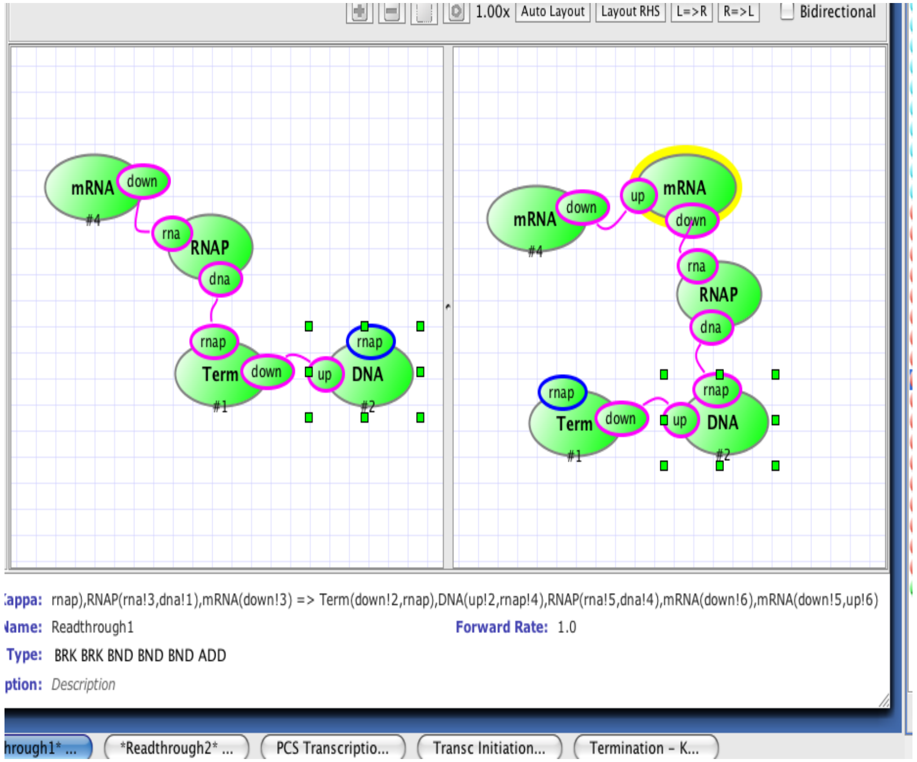
ption: *Description*

Readthrough 1



Forward rate: 10 s^-1

Kappa string: DNA(binding!1,downstream!2,type~BBaC0012), RNAP(dna!1,rna!3), DNA(upstream!2,binding), RNA(downstream!3) -> DNA(binding,downstream!2,type~BBaC0012), RNAP(dna!1,rna!3), DNA(upstream!2,binding!1), RNA(downstream!4), RNA(binding,upstream!4,downstream!3,type~BBaC0012)



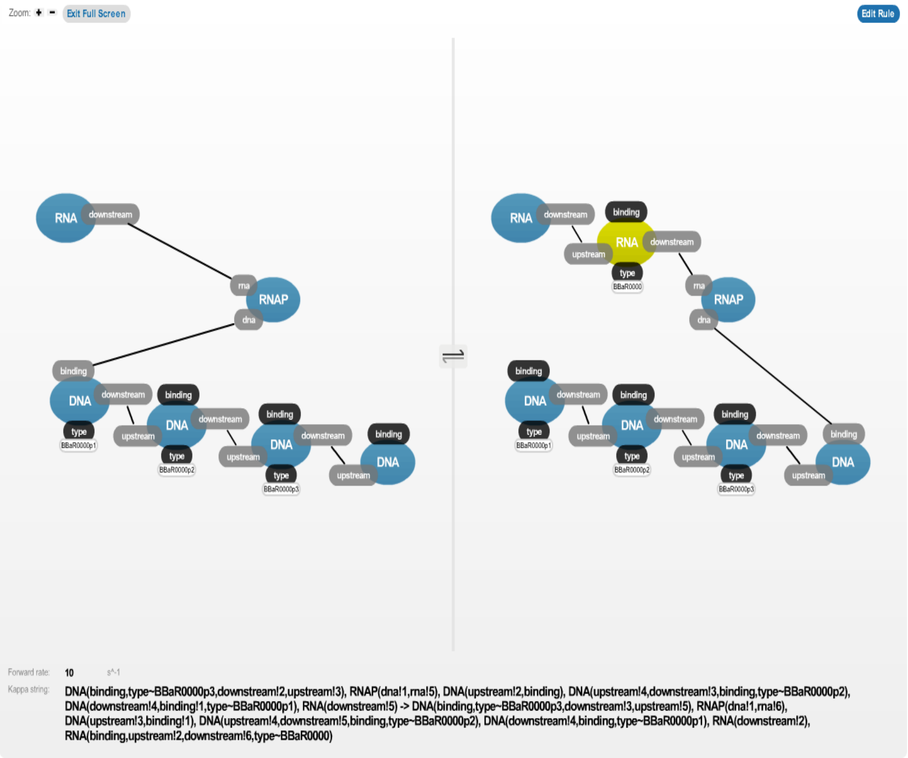
:appa: rnap),RNAP(rna!3,dna!1),mRNA(down!3) => Term(down!2,rnap),DNA(up!2,rnap!4),RNAP(rna!5,dna!4),mRNA(down!6),mRNA(down!5,up!6)

lame: Readthrough1 Forward Rate: 1.0

Type: BRK BRK BND BND BND ADD

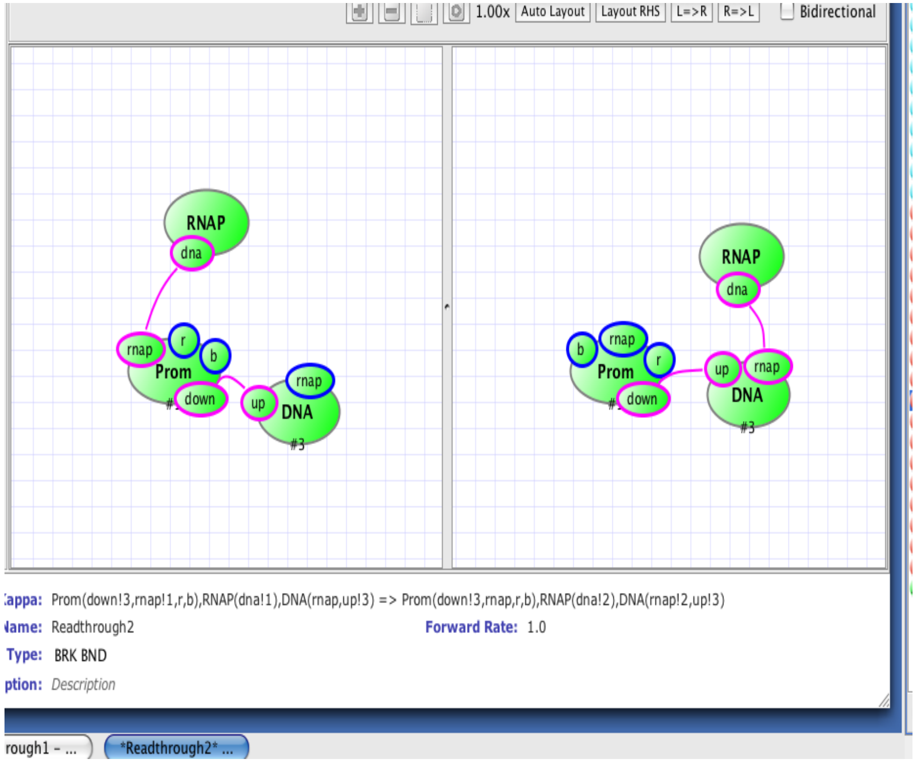
ption: *Description*

Readthrough 2



Forward rate: 10 s^-1

Kappa string: DNA(binding,type~BBaR0000p3,downstream!2,upstream!3), RNAP(dna!1,rna!5), DNA(upstream!2,binding), DNA(upstream!4,downstream!3,binding,type~BBaR0000p2), DNA(downstream!4,binding!1,type~BBaR0000p1), RNA(downstream!5) -> DNA(binding,type~BBaR0000p3,downstream!3,upstream!5), RNAP(dna!1,rna!6), DNA(upstream!3,binding!1), DNA(upstream!4,downstream!5,binding,type~BBaR0000p2), DNA(downstream!4,binding,type~BBaR0000p1), RNA(downstream!2), RNA(binding,upstream!2,downstream!6,type~BBaR0000)



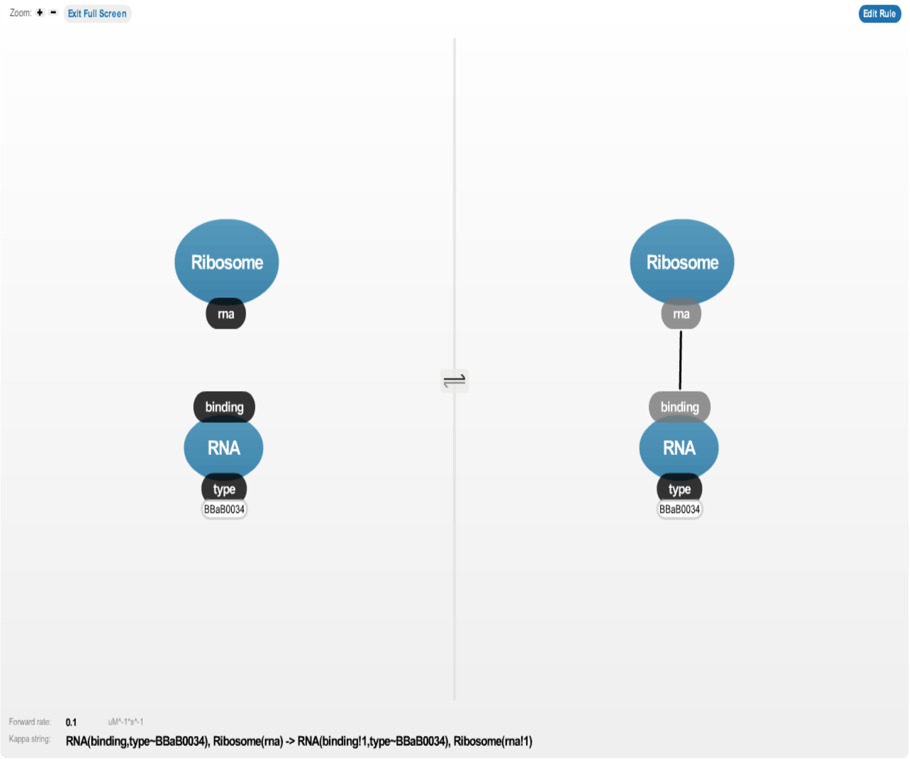
:appa: Prom(down!3,rnap!1,r,b),RNAP(dna!1),DNA(rnap,up!3) => Prom(down!3,rnap,r,b),RNAP(dna!2),DNA(rnap!2,up!3)

\lame: Readthrough2 **Forward Rate:** 1.0

Type: BRK BND

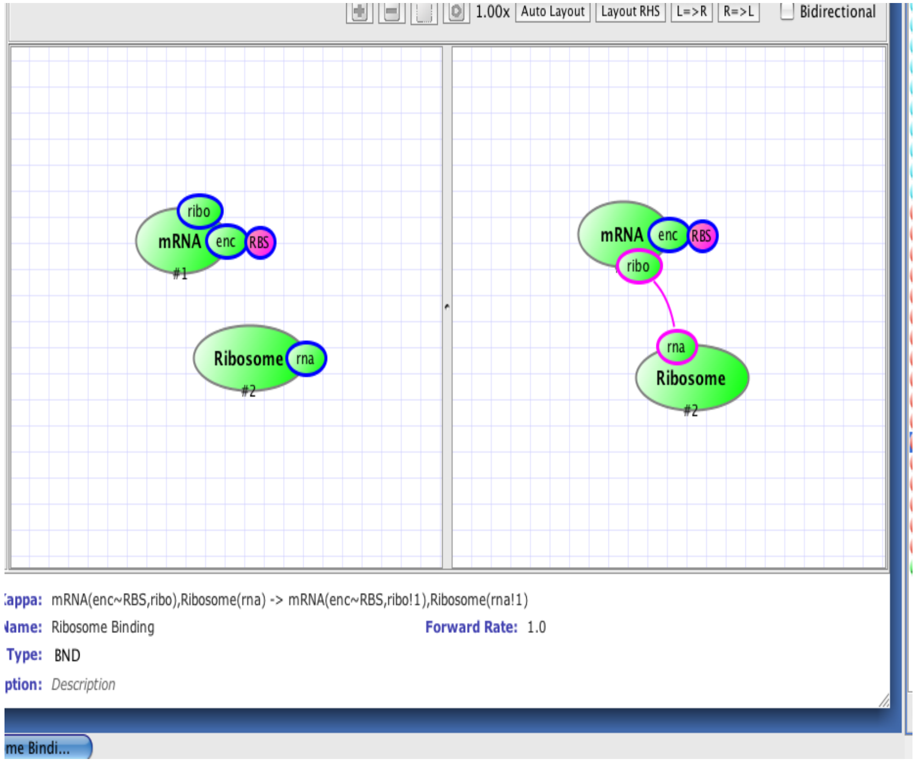
ption: *Description*

Ribosome Binding



Forward rate: 0.1 uM^-1*s^-1

Kappa string: RNA(binding,type~BBaB0034), Ribosome(rna) -> RNA(binding!1,type~BBaB0034), Ribosome(rna!1)



appa: mRNA(enc~RBS,ribo),Ribosome(rna) -> mRNA(enc~RBS,ribo!1),Ribosome(rna!1)

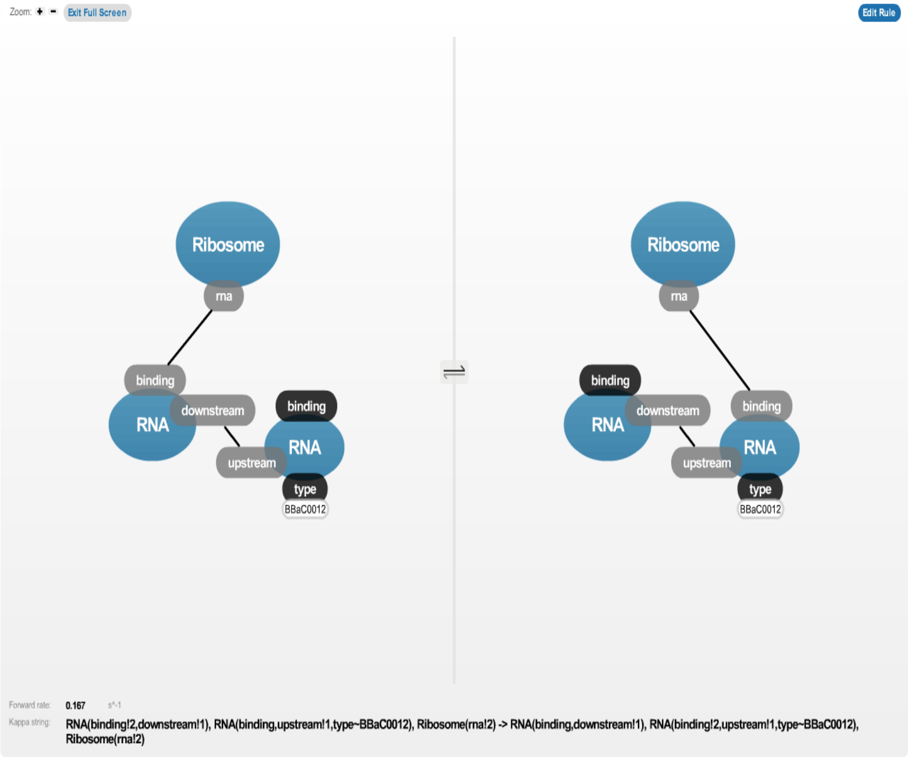
ame: Ribosome Binding

Forward Rate: 1.0

Type: BND

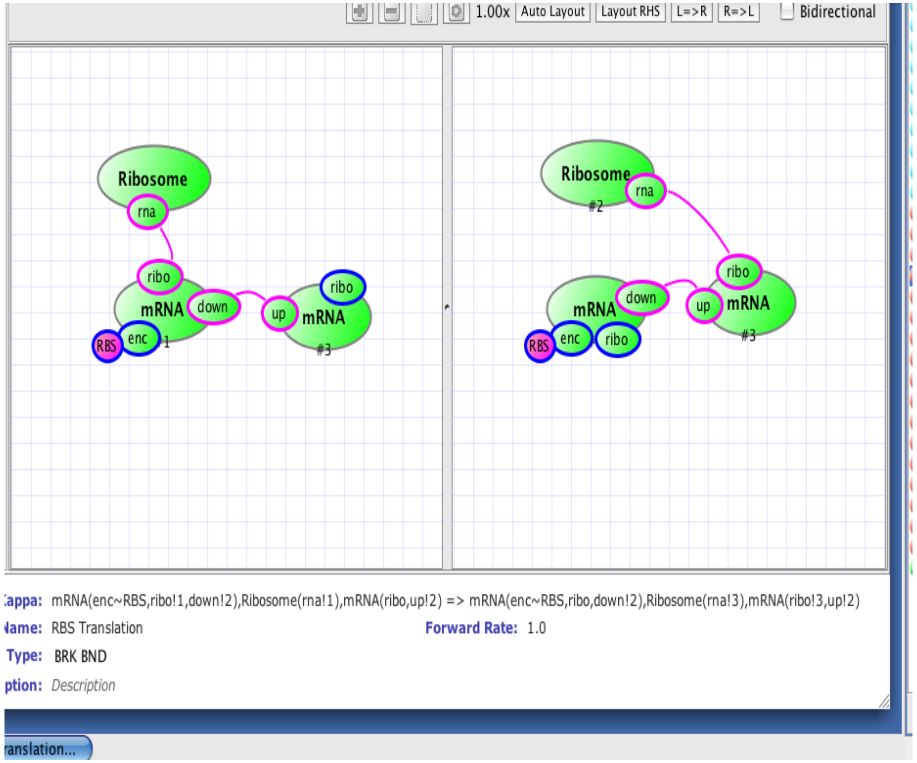
ption: *Description*

Translation Initiation



Forward rate: 0.167 s^-1

Kappa string: RNA(binding!2,downstream!1), RNA(binding,upstream!1,type~BBaC0012), Ribosome(rna!2) -> RNA(binding,downstream!1), RNA(binding!2,upstream!1,type~BBaC0012), Ribosome(rna!2)



:appa: mRNA(enc~RBS,ribo!1,down!2),Ribosome(rna!1),mRNA(ribo,up!2) => mRNA(enc~RBS,ribo,down!2),Ribosome(rna!3),mRNA(ribo!3,up!2)

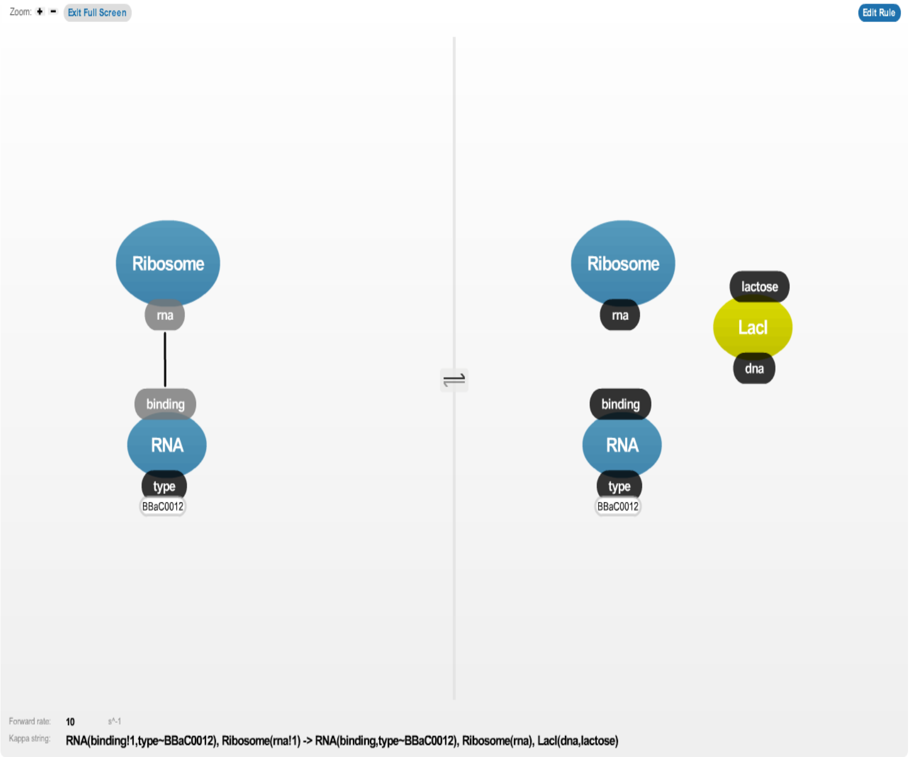
lame: RBS Translation

Forward Rate: 1.0

Type: BRK BND

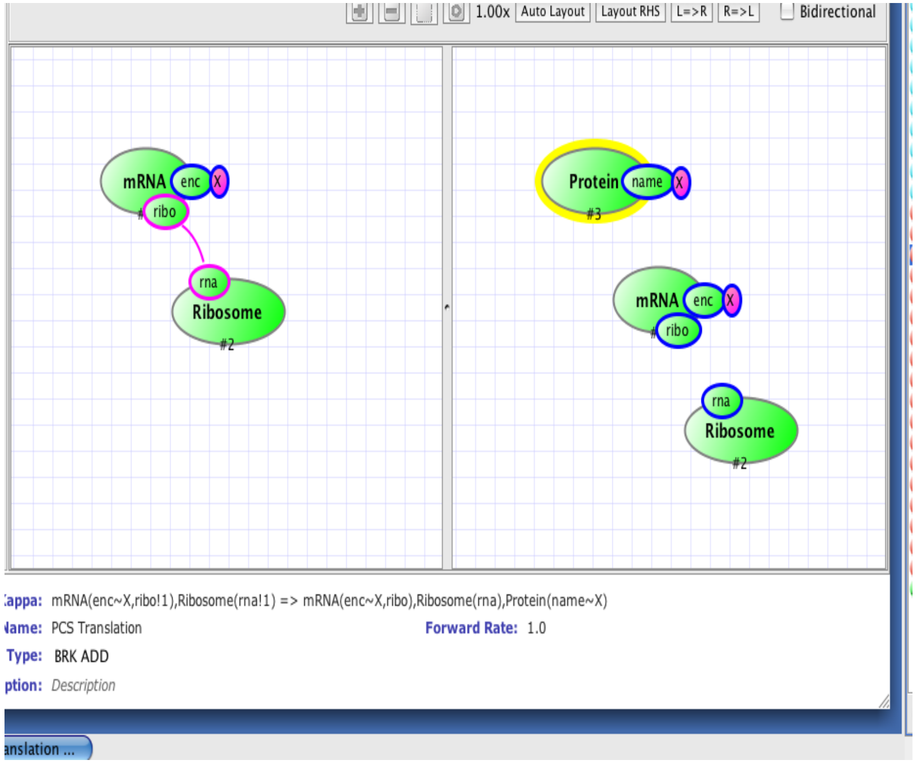
ption: *Description*

Translation



Forward rate: 10 s^-1

Kappa string: RNA(binding!1,type~BBaC0012), Ribosome(rna!1) -> RNA(binding,type~BBaC0012), Ribosome(rna), LacI(dna,lactose)



appa: mRNA(enc~X,ribo!1),Ribosome(rna!1) => mRNA(enc~X,ribo),Ribosome(rna),Protein(name~X)

lame: PCS Translation

Forward Rate: 1.0

Type: BRK ADD

ption: *Description*

Promoter Design

- In the previous rule set, the hierarchy extension did not add much to the model apart from readability of rules and initial conditions.

- In fact, if anything, they make the rules less generic as if we only had DNA agents we could write one generic transcription rule instead of three.

- However the hierarchy extension is very useful when it comes to designing promoters.

- The promoter in the example had only one binding site for repressors/activators. however it is very common for promoters to have multiple binding sites.

A Promoter with Multiple Binding Sites

$Prom1 = Prom[..., r = r1, r2, r3]$

- If we take the previous rule for binding a repressor
  $RepBind : Prom(r, b), Repressor(s)- > Prom(r^1, b), Repressor(s^1)$
- The probability that a rule fires at a given time point depends on the activitiy of the rule, where the activity is the rate of the rule * the number of way the rule can be applied to the current state of the system.
- The rule $RepBind$ will have an additive activity depending on how many of its binding sites are free.
- Using hierarchies provides a simple way to design complex promoter behaviour.