Applications of Rewriting Logic in Biology

II

Representation of Cellular Signaling in Maude

Carolyn Talcott
SRI International

July 2007

PLAN

- Biological Processes
- Symbolic systems biology
- Pathway Logic Representation
  - Representing cells
  - Representing Reactions
- A simple reaction network

Biological Processes & Symbolic Systems Biology

Biological Systems

- Biological processes are complex
  - genes, proteins, metabolites
  - cells, organs, organisms
- Dynamics that range over huge timescales
  - microseconds to years
- Spatial scales over 12 orders of magnitude
  - single protein to cell, cell to whole organism
- Oceans of experimental biological data generated
- Important intuitions captured in mental models that biologists build of biological processes

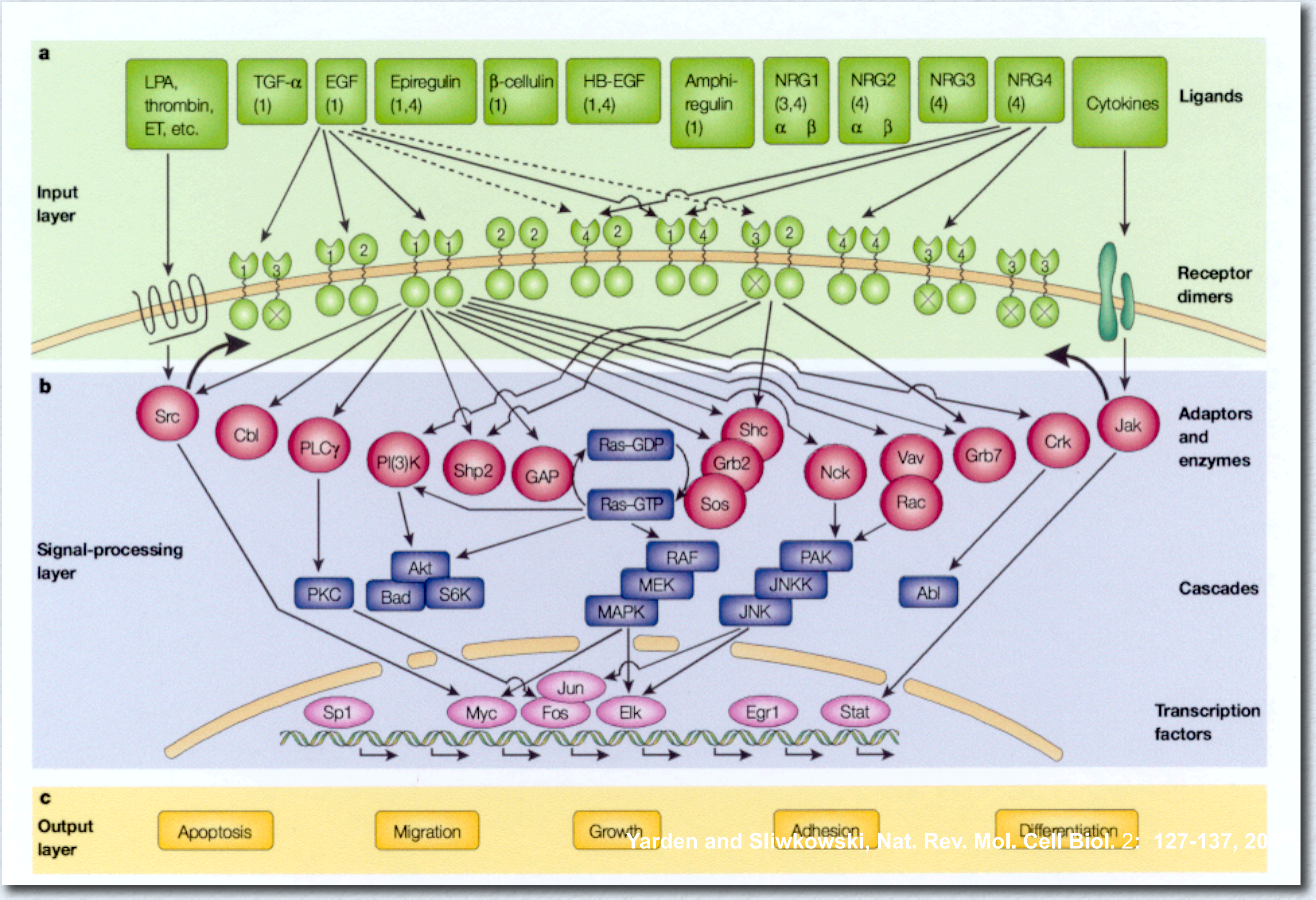
Cellular Signaling

- Cells respond to changes in their environment through biochemical pathways that detect, transduce, and transmit information to effector molecules within different cellular compartments.
- Most signaling pathways involve hierarchical assembly in space and time of multi-protein complexes that regulate the flow of information according to logical rules.
- Biological subnetworks interact to produce high levels of physiological organization (e.g., circadian clock subnetworks are integrated with metabolic, survival, and growth subnetworks).

Signaling Pathways

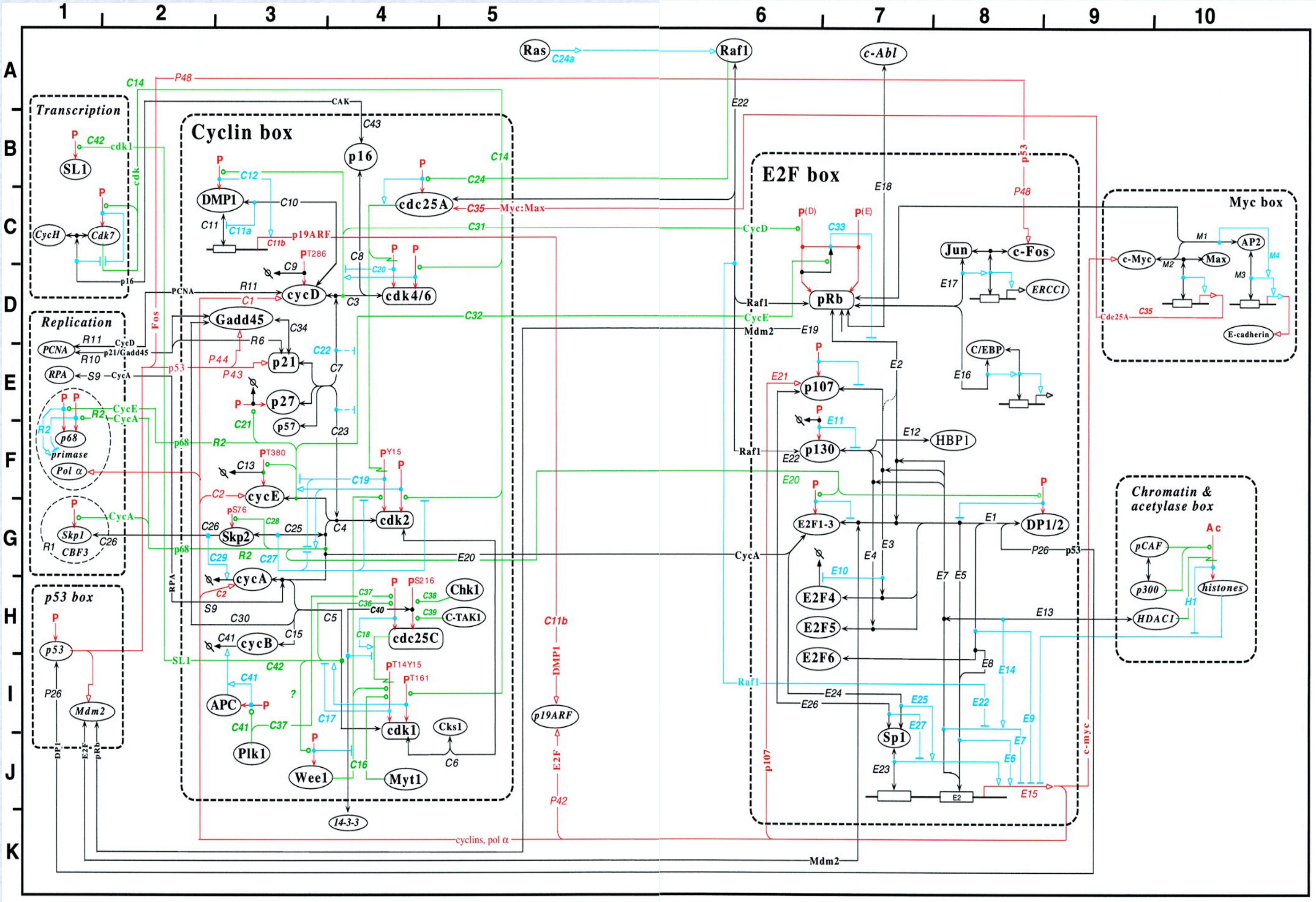
- Signaling pathways involve the modification and/or assembly of proteins and other molecules within cellular compartments into complexes that coordinate and regulate the flow of information.
- Signaling pathways are distributed in networks having stimulatory (positive) and inhibitory (negative) feedback loops, and other concurrent interactions to ensure that signals are propagated and interpreted appropriately in a particular cell or tissue.
- Signaling networks are robust and adaptive, in part because of combinatorial complex formation (several building blocks for forming the same type of complex), redundant pathways, and feedback loops.

The ErbB Network

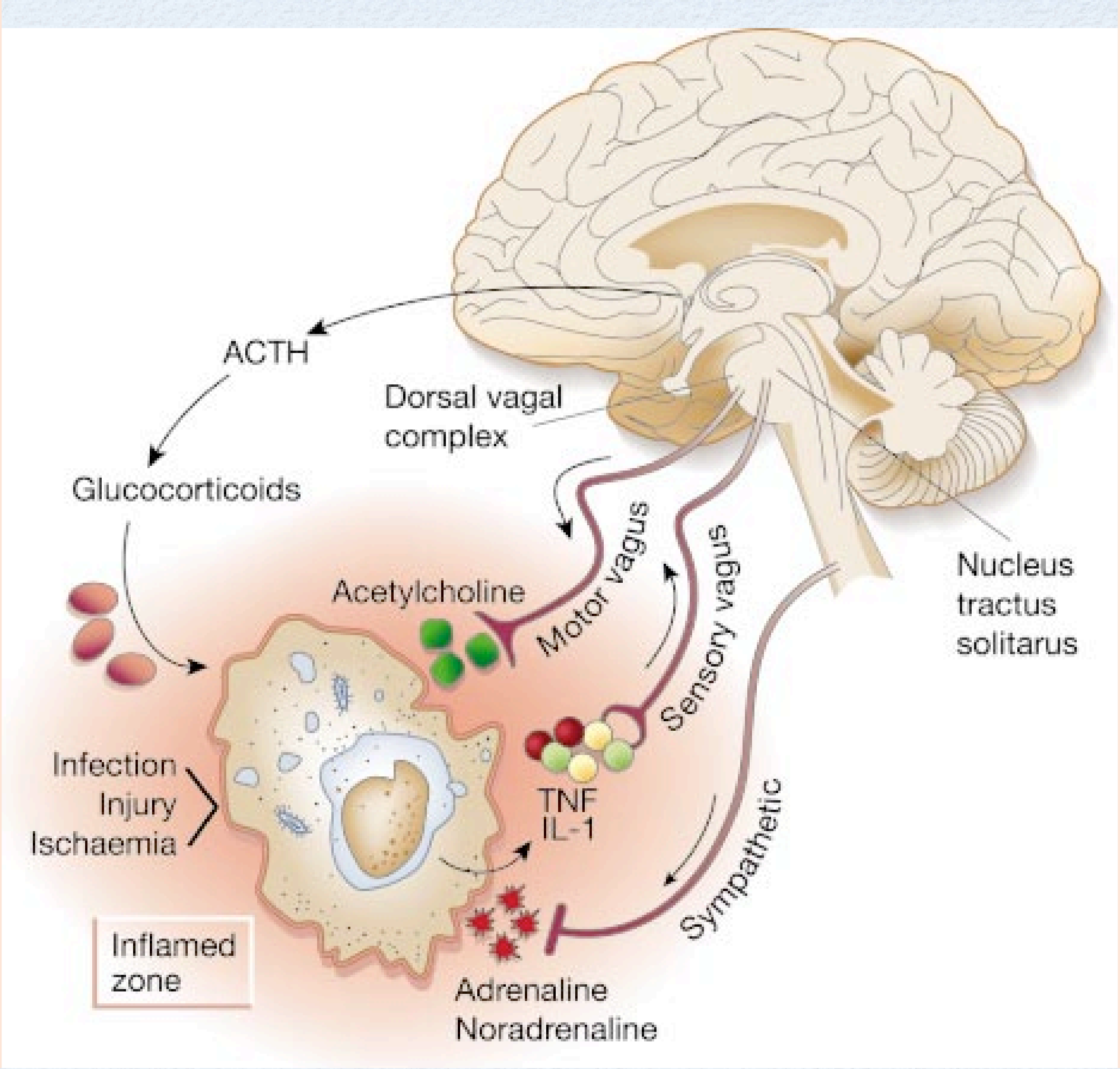


Yarden and Sliwkowski, Nat. Rev. Mol. Cell Biol. 2: 127-137, 20

Cell Cycle Control

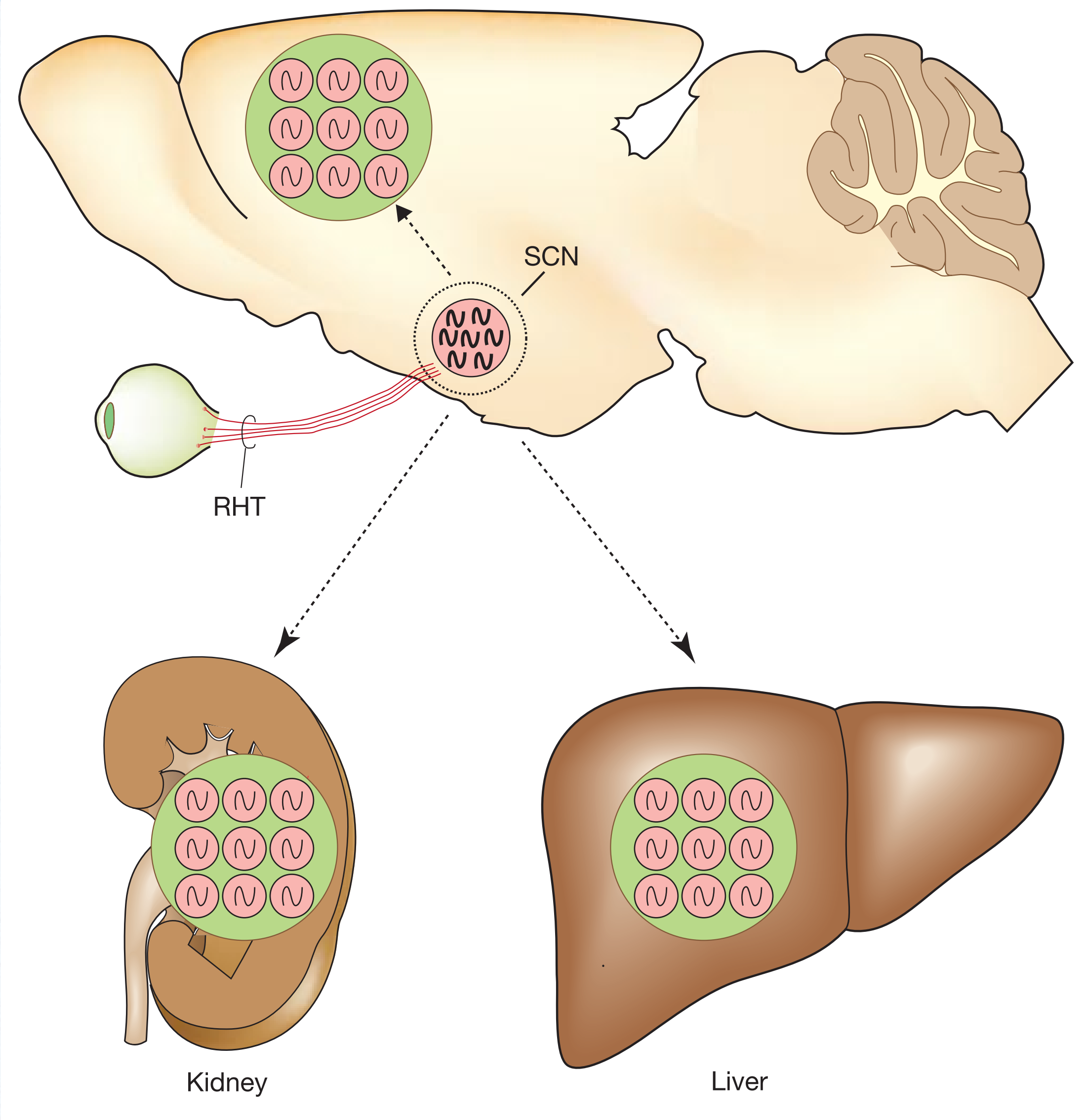


Inflamatory reflex



Kevin J Tracey
Nature Vol 420 2002

COORDINATION OF CIRCADIAN TIMING



S. M. Reppert & D. R. Weaver
Nature Vol 18 2002

Symbolic Systems Biology

The ***qualitative and*** quantitative study of biological processes as ***integrated*** systems rather than as isolated parts

Goals:

- Model causal networks of biomolecular interactions and reactions in a logical framework
- Develop formal models that are as close as possible to domain expert's mental models
- Compute with and analyze these complex networks
  - Abstract and refine logical models
  - Simulate or use deduction to check properties
  - Make predictions about possible outcomes, experiment, update model

Formally Based Systems
A Sampling

- Pathway Logic
- BIOCHAM
- Membrane calculi -- spatial process calculi / logics
    - Brane calculus -- mobility of membranes
    - P Systems -- mobility of processes
- Statecharts
- BioSPI, SPIM -- stochastic pi
- Hybrid SAL -- hybrid (discrete + continuous) systems

ABOUT PATHWAY LOGIC

Pathway Logic (PL) is an approach to modeling biological processes as executable formal specifications (in Maude)

The resulting models can be queried

- using formal methods tools: given an initial state
  - execute --- find some pathway
  - search --- find all reachable states satisfying a given property
  - model-check --- find a pathway satisfying a temporal formula
- using reflection
  - find all rules that use / produce X (for example, activated Rac)
  - find rules down stream of a given rule or component

Pathway Logic Team

- Keith Laderoute
- Patrick Lincoln
- Carolyn Talcott

- Linda Briesemeister
- Steven Eker
- Merrill Knapp
- Ian Mason
- Andy Poggio
- Malabika Sarker
- Ashish Tiwari

- Biology Computer Science

Pathway Logic (PL) Representation of Signaling

http://pl.csl.sri.com/

Pathway Logic Organization

A Pathway Logic (PL) knowledge base has four parts

- Theops --- sorts and operations
- Components --- specific proteins, chemicals ...
- Rules --- signal transduction reactions
- Dishes --- candidate initial states

A PL cell signaling model is generated from

- a knowledge base
- a dish

THEOPS: OVERVIEW

Specifies sorts and operations (data types) used to represent cells:

- Proteins and other compounds
- Complexes
- Soup --- mixtures / solutions / supernatant ...
- Post-translational modifications
- Locations --- cellular compartments refined
- Cells --- collection of locations
- Dishes --- for experiments, think Petri dish

THEOPS: SOUP

```
fmod PROTEIN is inc NAT .
  sorts AminoAcid Protein .
  subsort AminoAcid < Protein .
  ... endfm

fmod THING is inc PROTEIN .     *** Basic building blocks
  sort Thing Family Composite Complex Chemical Signature .
  subsorts Protein Family Composite Complex Chemical Signature
          < Thing .
  op (_:_) : Thing Thing -> Complex [assoc comm] .
  ... endfm
```

Example complex: 1433x1 : [Raf1 - phos(S 259)]

```
fmod SOUP is inc THING .         *** mixtures
  sort Soup .                   *** a multiset of Thing
  subsort Thing < Soup .
  op empty : -> Soup .
  op __ : Soup Soup -> Soup [assoc comm id: empty ] .

  op _has_ : Soup Thing -> Bool .
  eq (T1:Thing S:Soup ) has T2:Thing =
     if T1:Thing == T2:Thing
     then true else S:Soup has T2:Thing fi .
  eq (S:Soup has T:Thing) =  false [owise] .
  ... endfm
```

Example soup: Src Shc 1433x1 Raf1

Theops: Modifications & Locations

```
fmod MODIFICATION is pr SOUP .
  sorts  Modification ModSet .    *** multisets of modifications
  subsort Modification < ModSet .
  op none : -> ModSet .
  op __ : ModSet ModSet -> ModSet [assoc comm id: none] .

  op [_-_] : Protein   ModSet -> Protein [right id: none ] .

  op act : -> Modification .
  op bound :       -> Modification .
  op phos :        -> Modification .
  op Yphos : -> Modification .
  ops GTP GDP : -> Modification .    *** used for small GTPases
endfm
```

Examples: [EgfR - act]   [Cbl - Yphos] [Hras - GDP]   [Egf - bound]

```
fmod LOCATION is inc MODIFICATION .
sort Location LocName .
subsort Location < Soup .
op {_|_} : LocName Soup -> Location [format (n d d t d d)] .

ops CLo CLm CLi CLc : -> LocName .   *** Cell - out,mem,in,cytosol
ops NUo NUm NUi NUc : -> LocName .   *** Nucleus - out,mem,in,cytosol
...
endfm
```

THEOPS: CELLS & DISHES

```
mod CELL is inc LOCATION .
  sorts Cell CellType .
  subsort Cell < Soup .
  op [_|_] : CellType Soup -> Cell .
  op Cell : -> CellType .
  op HMEC : -> CellType .
  ...
endm
```

Example cell:

```
    [HMEC | {CLm | EgfR PIP2}{CLi | [Hras - GDP] Src}
            {CLc | Cbl Gab1 Grb2 Pi3k Plcg Shc Sos1 Vav2}]
```

```
mod DISH is inc CELL .
  sort Dish .
  op PD : Soup -> Dish .
endm
```

Example dish:

```
 PD(Egf [HMEC | {CLm | EgfR PIP2}{CLi | [Hras - GDP] Src}
                {CLc | Cbl Gab1 Grb2 Pi3k Plcg Shc Sos1 Vav2}])
```

Components

```
sort ErbB1L . subsort ErbB1L < Protein . *** ErbB1 Ligand

op Egf : -> ErbB1L [metadata "(\
  (spname EGF_HUMAN)\
  (spnumber P01133)\
  (hugosym EGF)\
  (category Ligand)\
  (synonyms \"Pro-epidermal growth factor precursor, EGF\" \
            \"Contains: Epidermal growth factor, Urogastrone \"))"] .

op EgfR : -> Protein [metadata "(\
  (spname EGFR_HUMAN)\
  (spnumber P00533)\
  (hugosym EGFR)\
  (category Receptor)\
  (synonyms \"Epidermal growth factor receptor precursor\" \
            \"Receptor tyrosine-protein kinase ErbB-1, ERBB1 \"))"] .

op Sos1  : -> Protein [metadata "(\
  (spname SOS1_HUMAN)\
  (spnumber Q07889)\
  (hugosym SOS1)\
  (synonyms \"Son of sevenless protein homolog 1, SOS-1 \" \
            \"SOS1 guanine nucleotide exchange factor \" \
            \"Son of sevenless, drosophila homolog 1  \"))"] .
```

RULES

- A PL rule specifies the change in a cell due to an enabled reaction. The rule label gives a hint as to what happens.
- In addition rules must be annotated with evidence
  - literature citations

    pubmed id (type: review, data) brief description
  - curator notes

RULE 1: RECEPTOR BINDING

A simplified description of the activation of EgfR:

If a dish contains an EgfR ligand (?ErbB1L:ErbB1L) outside a cell with EgfR in the cell membrane then the ligand binds to exterior part of the receptor and the receptor is activated.

```
rl[1.EgfR.on]: ?ErbB1L:ErbB1L
  [CellType:CellType | ct
  {CLo | clo                    }
  {CLm | clm EgfR               } ]
  =>
  [CellType:CellType | ct
  {CLo | clo [?ErbB1L:ErbB1L - bound] }
  {CLm | clm [EgfR - act]       } ] .
  -------------------------------------
  *** 11566606(R) ErbB1Ls are AR Egf TGFa Btc Epr Hbegf
  *** 12620237(D) Crystal structure of Egf-EgfR interaction.
  … …
```

RULE 5: RECRUITMENT

Activated EgfR recruits Grb2 to the inside of the cell membrane

```
rl[5.Grb2.reloc]:
  {CLm | clm [EgfR - act]      }
  {CLi | cli                   }
  {CLc | clc Grb2              }
  =>
  {CLm | clm [EgfR - act]      }
  {CLi | cli [Grb2 - reloc]    }
  {CLc | clc                   } .
  --------------------------------
  *** 2803273(D) Grb2 binds to EgfR-Grb2 pY1086
  … …
```

Rule 15: Phosphorylation

Activated EgfR recruits Cbl and phosphorylates it on a tyrosine

```
rl[15.Cbl.reloc.Yphos]:
  {CLm | clm [EgfR - act]   }
  {CLi | cli                }
  {CLc | clc Cbl            }
  =>
  {CLm | clm [EgfR - act]   }
  {CLi | cli [Cbl - Yphos] }
  {CLc | clc                } .
  ---------------------------
  *** 8662998(D) EgfR-Yphosed recruits Cbl to the CLi and
    *** phosphorylates it on Tyrosine
  ... ...
```

A small KB
Very early EGF stimulation

THE SOURCES

(See the directory SmallKB)

The four parts of the KB

```
theops.maude
components.maude
rules.maude
qq.maude
  rasDish --- Hras activation (picture in Pics/rasNet.pdf)
  rafDish  --- on to Raf1 etc.(picture in Pics/rafNet.pdf)
```

Additional code for analysis

```
pl-aux.maude
qqq-mc.maude
qqq-pn.maude
```

Runing examples

`load-small-mc.maude` (using Maude rules)

`load-small-pn.maude` (using Petri net form)

rasNet

Rule instances relevant to Hras activation

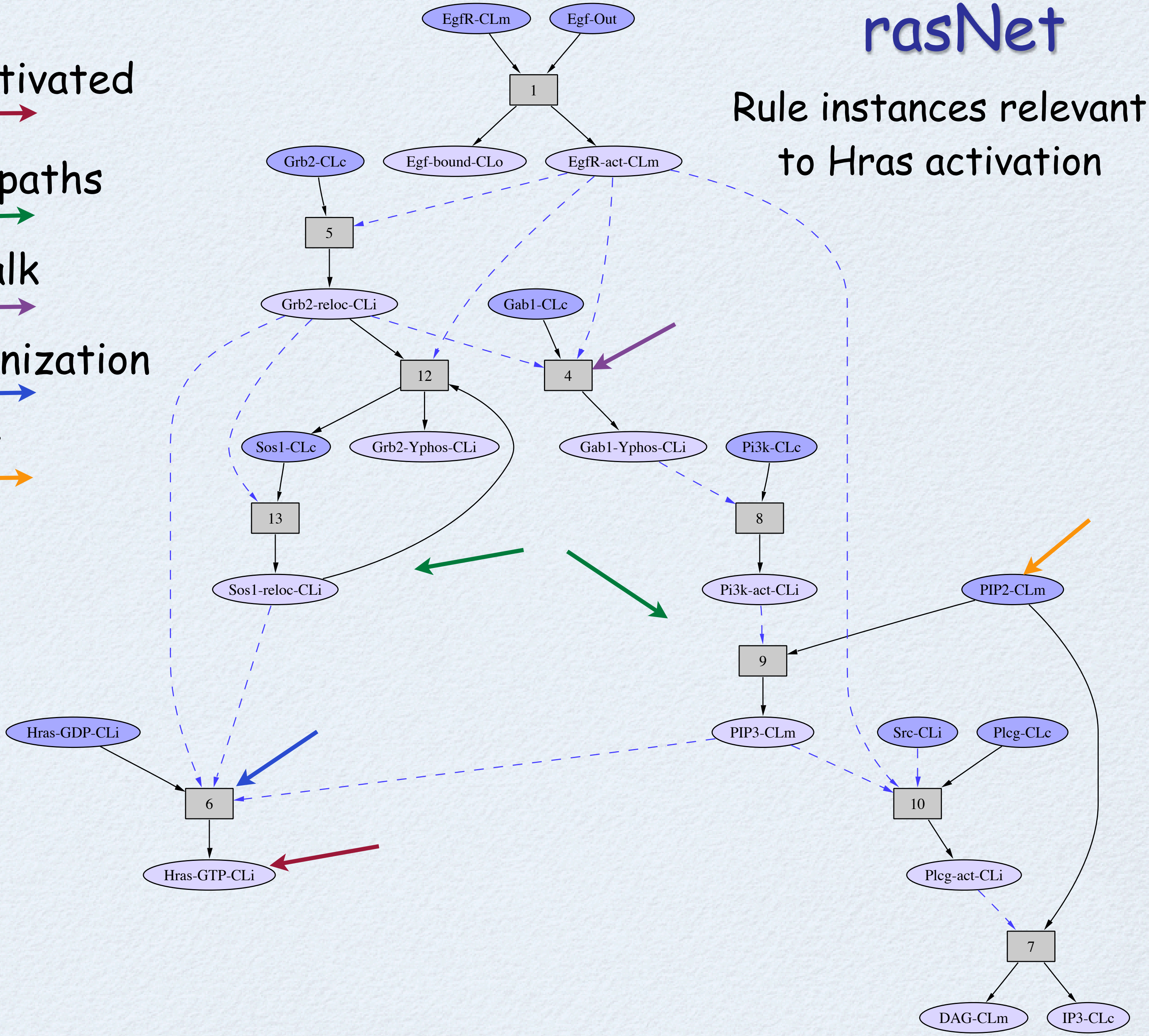
Hras activated

Parallel paths

Cross talk

Synchronization

Conflict



WHERE DID THE GRAPH COME FROM?

(See part iii for details)

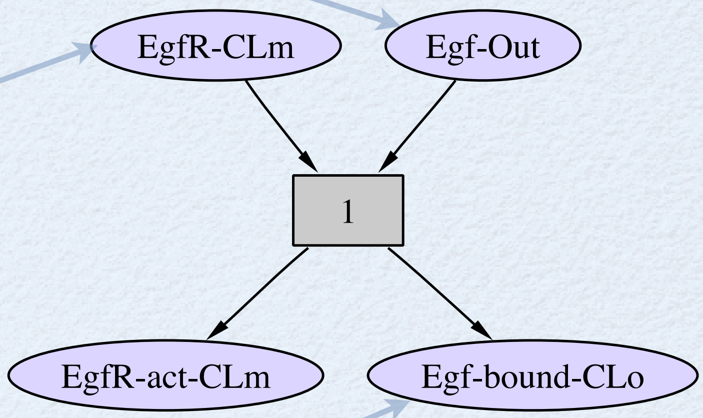
A rule can be represented as a rule node with lhs/rhs represented as incoming/ outgoing `occurrence' nodes.

A occurrence label represents a components modifications and location.

EgfR-CLm represents EgfR in the CLm location.

EgfR-act-CLm represents activated EgfR ([EgfR - act]) in the CLm location.

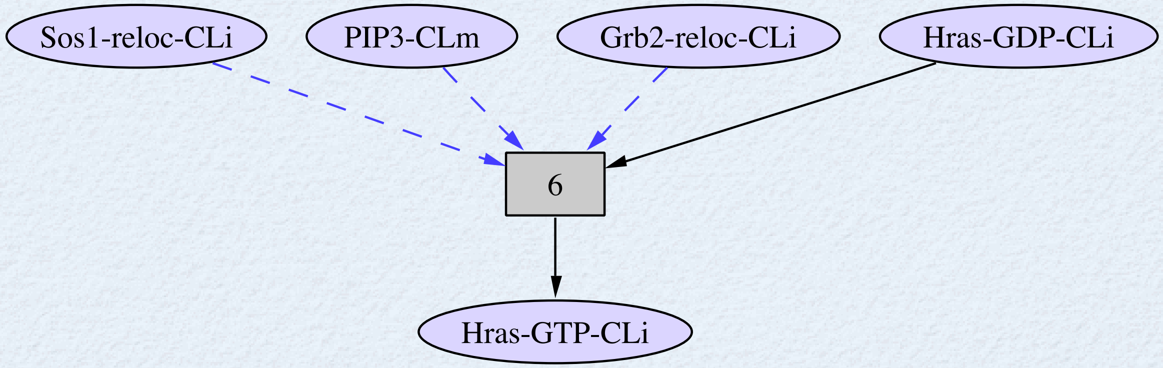
```
rl[1.EgfR.act]:
  Egf
  [CellType:CellType | ct
  {CLo | clo               }
  {CLm | clm EgfR          } ]
  =>
  [CellType:CellType | ct
  {CLo | clo [Egf - bound] }
  {CLm | clm [EgfR - act]  } ] .
```



Rules with Modifiers

A rule element that is the same on the left and right hand sides is represented using one occurrence node connected to the rule with a dashed arrow.

```
rl[6.Hras.act.1]:
  {CLm | clm PIP3                                       }
  {CLi | cli [Grb2 - reloc] [Sos1 - reloc] [Hras - GDP] }
  =>
  {CLm | clm PIP3                                       }
  {CLi | cli [Grb2 - reloc] [Sos1 - reloc] [Hras - GTP] } .
```



Computing with Rasdish

Find one possible execution using the rewrite command. Maude repeats the command (elaborated), prints statistics, and prints the final term.

```
Maude> rew rasDish .
rewrite in QQQ-MC : rasDish .
rewrites: 9 in 0ms cpu (2ms real) (~ rewrites/second)
result Dish: PD([HMEC |
{CLo |  [Egf - bound]}
{CLm |  PIP3 [EgfR - act]}
{CLi |  Src [Gab1 - Yphos] [Grb2 - Yphos]
       [Hras - GDP] [Pi3k - act] [Plcg - act]}
{CLc |  Sos1}])
```

If you suspect rewriting will not terminate use the bounded rewrite command, possibly followed continuation commands .

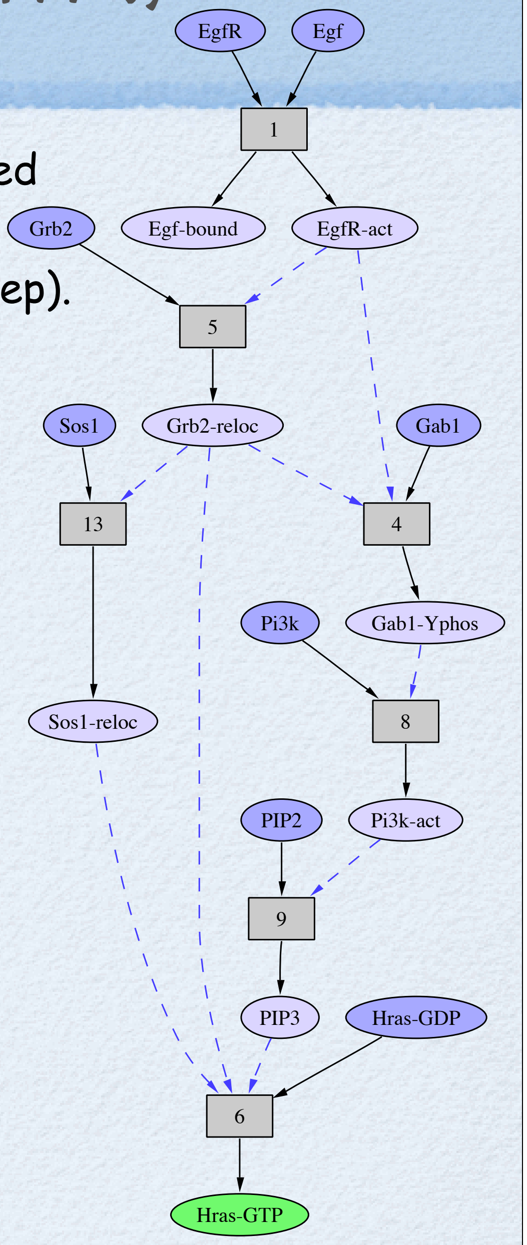
```
Maude> rew [n] rasDish .
Maude cont m .
```

Searching for a pathway

To find a pathway (computation) leading to activation of Hras (loaded with GTP) one can use the search command with a suitable search pattern and parameters ([1] -- the first solution, =>+ at least one step). Then ask Maude for the rule labels.

```
Maude> search [1] rasDish =>+
  PD(out:Soup [HMEC | cyto:Soup
              {CLi | cli:Soup [Hras - GTP]}]) .
Solution 1 (state 15)
out:Soup --> empty
cyto:Soup --> {CLo |[Egf - bound]}
              {CLm | PIP3 [EgfR - act]}
              {CLc | Plcg}
cli:Soup --> Src[Gab1 - Yphos][Grb2 - reloc]
             [Pi3k - act][Sos1 - reloc]

Maude> show path labels 15 .
1.EgfR.act
5.Grb2.reloc
4.Gab1.Yphosed
8.Pi3k.act
9.PIP3.from.PIP2.by.Pi3k
13.Sos1.reloc
6.Hras.act.1
```



Searching for other pathways

- Other pathways include
  - activation of both Hras and Plcg
  - restoring Sos1 after activation of Hras
- There is no pathway leading to IP3
  - Can you explain why?

Finding Pathways using Model-Checking: I

First we define some properties: goalP and avoidP.
In dish D, goalP holds of a thing th and a location l
if th occurs in D in location l (hasOcc(D,th,l).

```
op hasOcc : Dish Thing Loc -> Bool .
eq hasOcc(PD(out:Soup
             [ct:CellType | s:Soup {l:LocName | ls:Soup th:Thing}]),
          th:Thing, l:LocName) = true .
eq hasOcc(PD(out:Soup th:Thing [ct:CellType | s:Soup]), th:Thing, Out)
    = true .
eq hasOcc(PD(s:Soup), th:Thing, l:Loc) = false [owise] .

ops goalP avoidP : Thing Loc -> Prop .
eq PD(s:Soup) |= goalP(th:Thing, l:Loc)
    = hasOcc(PD(s:Soup),th:Thing,l:Loc) .
eq PD(s:Soup) |= avoidP(th:Thing, l:Loc)
    = not(hasOcc(PD(s:Soup),th:Thing,l:Loc)) .
```

To find a pathway leading to a given goal (<> Goal), model check the negation ([] ~ Goal) and if there is a counter example, the list of rules gives such a pathway.

Finding pathways using model-checking: II

To find a pathway activating Hras, use the goal:

```
goalP([Hras - GTP],CLi).
```

```
Maude> red mc2Rules(modelCheck(rasDish,[]~ goalP([Hras - GTP],CLi))) .
rewrites: 65 in 0ms cpu (3ms real) (~ rewrites/second)
result RuleNameList:
    '1.EgfR.act '5.Grb2.reloc '4.Gab1.Yphosed '8.Pi3k.act
    '9.PIP3.from.PIP2.by.Pi3k '10.Plcg.act '13.Sos1.reloc '6.Hras.act.1
    '12.Sos1.reinit deadlock
```

Note that the counterexample has more rules than the pathway found by search. This is because

1. search is breadth-first and model-checking is depth first thus search will generally find shorter paths
2. a proper counter-example must be infinite, thus all rules that can fire must be given a chance.

Finding Pathways Using Model-Checking: III

So what can we do with model checking that we can't do (better) with search? We can specify properties of the path not just a reachable state.

For example, search and model-checking both gave paths in which Gab1 activation (rule 4) happens before Sos1 relocation (rule 13). Is there a dependency or is there a pathway activating Hras, in which Sos1 is relocated before Gab1 is activated?

To find out we formalize this property and model-check its negation.

```
Let P := goalP([Gab1 - Yphos], CLi)
    Q := goalP([Sos1 - reloc], CLi)
    R := goalP([Hras - GTP], CLi)
```

Then the formulas `(<> R /\ ~ P U Q)` and `(<> ~ P /\ Q /\ <> R)` both describe paths with the desired property.

Finding Pathways using Model-Checking: IV

Model checking `~(<> R /\ ~ P U Q)` we get a pathway with rule 13 before rule 4.

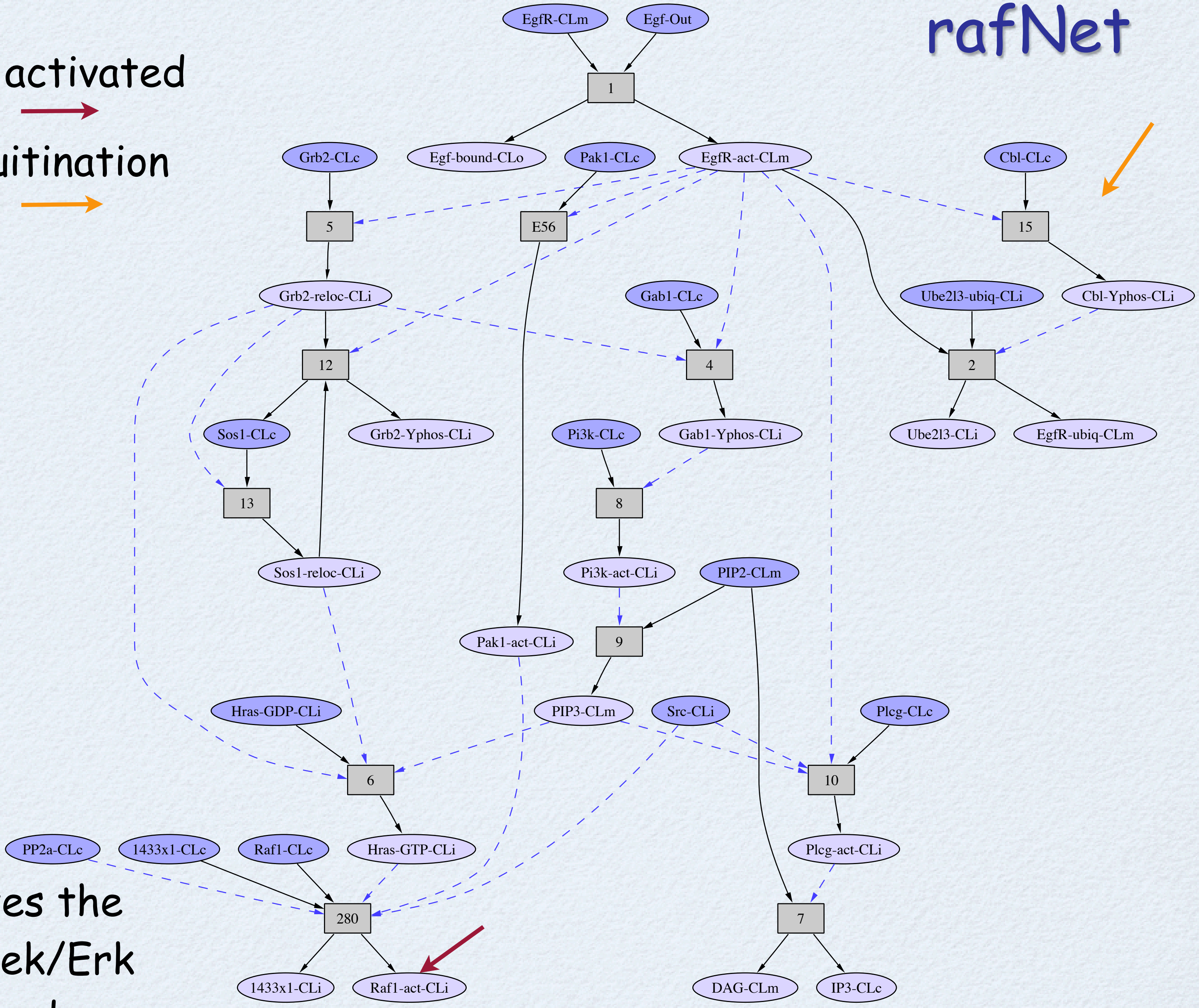
```
Maude> red mc2Rules(modelCheck(rasDish, ~ ( (<> goalP([Hras - GTP],
CLi)) /\ (~ goalP([Gab1 - Yphos], CLi) U goalP([Sos1 - reloc],
CLi))))) .
rewrites: 73 in 0ms cpu (4ms real) (~ rewrites/second)
result RuleNameList:
    '1.EgfR.act '5.Grb2.reloc '13.Sos1.reloc '4.Gab1.Yphosed
    '8.Pi3k.act '9.PIP3.from.PIP2.by.Pi3k '6.Hras.act.1 '10.Plcg.act
    '12.Sos1.reinit deadlock
```

Model checking `~( <> (~ P /\ Q) /\ <> R))` results in the same pathway.

rafNet

Raf1 activated

Ubiquitination



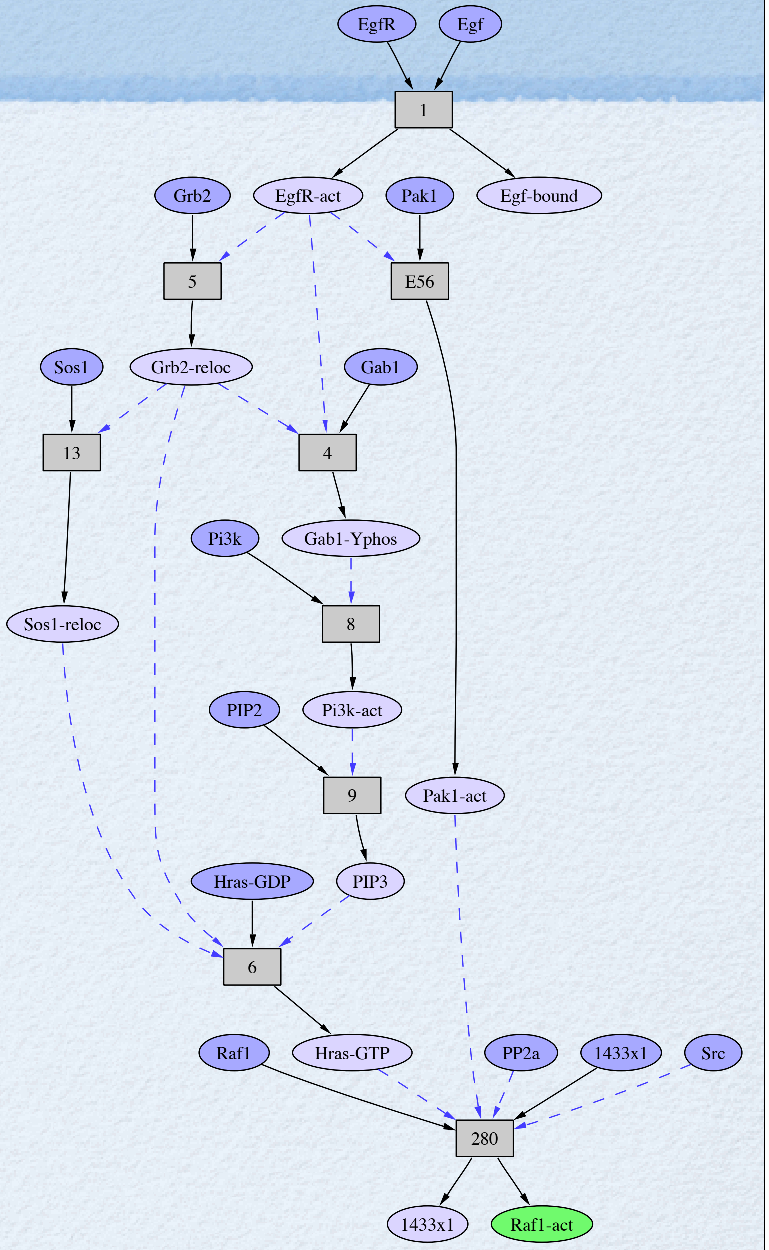
Initates the Raf/Mek/Erk cascade

Pathways to Activation of Raf1

by Search

```
Maude> search [1] rafDish =>+
    PD(out:Soup [HMEC | cyto:Soup
{CLi | cli:Soup [Raf1 - act]}]) .

Solution 1 (state 100)
out:Soup --> empty
cyto:Soup --> {CLo | [Egf - bound]}
              {CLm | PIP3 [EgfR - act]}
              {CLc | Cbl Plcg PP2a}
cli:Soup --> 1433x1 Src [Gab1 - Yphos]
             [Grb2 - reloc][Hras - GTP]
             [Pak1 - act][Pi3k - act]
             [Sos1 - reloc] [Ube2l3 - ubiq]
Maude> show path labels 100 .
1.EgfR.act
5.Grb2.reloc
4.Gab1.Yphosed
8.Pi3k.act
9.PIP3.from.PIP2.by.Pi3k
13.Sos1.reloc
6.Hras.act.1
E56.Pak1.irt.Egf
280.Raf1.by.Hras
```



Pathways to Activation of Raf1
by Model-Checking

The path found by search with some extra stuff.

```
Maude> red mc2Rules(modelCheck(rafDish,[] ~goalP([Raf1 - act], CLi))) .
result RuleNameList:
     '1.EgfR.act '5.Grb2.reloc '4.Gab1.Yphosed '8.Pi3k.act
     '9.PIP3.from.PIP2.by.Pi3k '10.Plcg.act '13.Sos1.reloc '6.Hras.act.1
     '12.Sos1.reinit '15.Cbl.reloc.Yphos 'E56.Pak1.irt.Egf '2.EgfR.ubiq
     '280.Raf1.by.Hras deadlock
```

The path found by forcing Gab1 activation before Sos1 relocation.

```
Maude> red mc2Rules(modelCheck(rafDish,
    ~((<> goalP([Raf1 - act], CLi)) /\
      (~ goalP([Gab1 - Yphos], CLi) U goalP([Sos1 - reloc], CLi))))) .
result RuleNameList:
     '1.EgfR.act '5.Grb2.reloc '13.Sos1.reloc '4.Gab1.Yphosed
     '8.Pi3k.act '9.PIP3.from.PIP2.by.Pi3k '6.Hras.act.1 '10.Plcg.act
     '12.Sos1.reinit '15.Cbl.reloc.Yphos 'E56.Pak1.irt.Egf '2.EgfR.ubiq
     '280.Raf1.by.Hras deadlock
```

Exercises

- In directory SmallKB
- README.txt summarizes file contents
- exercises.txt contains instructions and problems to solve.
- answers.txt cantian some answers