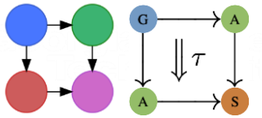
Compositional exploration of combinatorial models

Kris Brown, Tyler Hanks, James Fairbanks

20.07.22

Outline



UF

Introduction

- Why represent scientific models combinatorially?

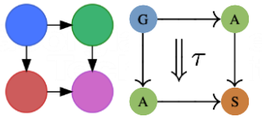
Model space exploration

- The category of diagrams as a category of model spaces
- Example limits and colimits
- Composition recipes
- Limits and colimits: implemenation

Model Selection

- Best fit chemical reaction network example

Alternative: model as opaque code / math / logic



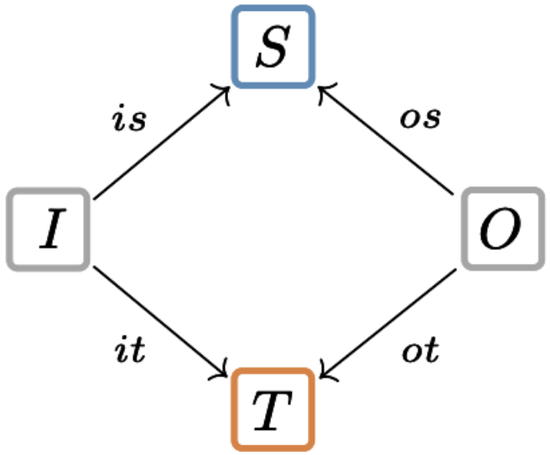
UF

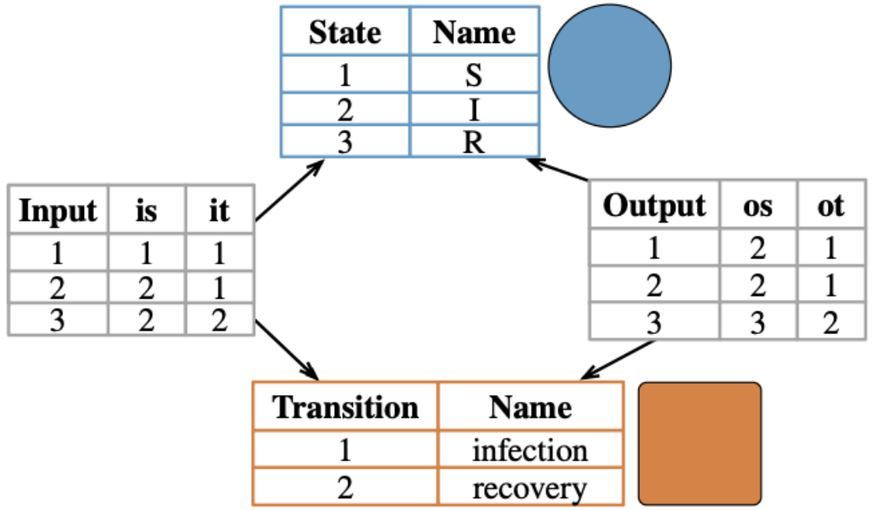
```
"""
2 H₂ + O₂ → 2 H₂O. Mass-action kinetics. Compare to experimental data and plot.
"""
def main():
    # experimental data
    real_data = [0.0101, 0.012, 0.023, 0.037, 0.045, 0.053, 0.061, 0.069,
    0.076, 0.083, 0.089, 0.096, 0.102, 0.108, 0.114, 0.119, 0.125, 0.130, 0.135,
    0.140, 0.145, 0.150, 0.154, 0.159, 0.163, 0.167, 0.171, 0.175, 0.179, 0.183,
    0.186, 0.190, 0.193, 0.197, 0.200, 0.203, 0.206, 0.209, 0.212, 0.215, 0.218,
    0.221, 0.224, 0.227, 0.229, 0.232, 0.234, 0.237, 0.239]
    # Initial concentrations
    H2,O2, H2O = 1.0, 2.0, 0.0
    dt = 0.01
    results = []
    for step in range(1,50):
        print("Step ", step)
        rate = 0.5 * H2**2 * O2
        H2 -= 2*rate*dt
        O2 -= rate*dt
        H2O += rate*dt
        results.append(H2O)

    plot(results, real_data) # Figure 4 in the paper
```

Many tasks we'd like to do cannot be done with arbitrary code (nor mathematical expressions).

1. Generate the entire code from just declaring the reaction
2. Easily alter semantics (e.g. stochastic-based simulation)
3. Construct models compositionally (operad)
4. Construct models compositionally (limits).
5. Explore alternate reaction networks to fit the data

1. Functorial generation of code



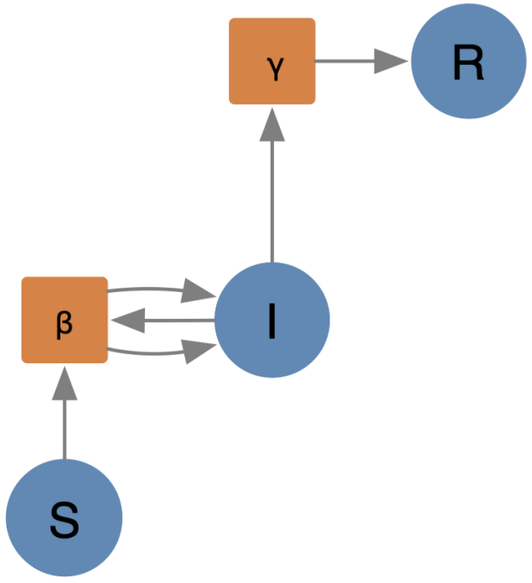


State	Name
1	S
2	I
3	R

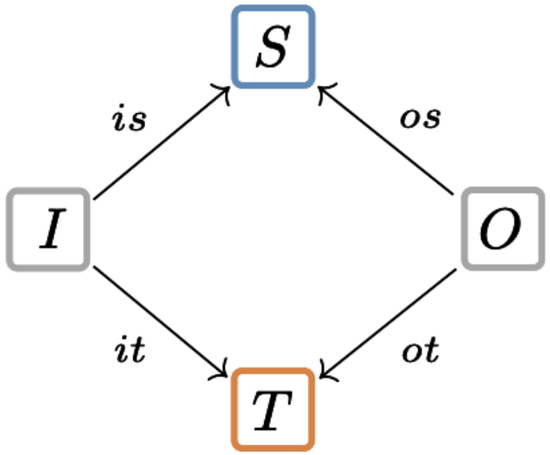
Input	is	it
1	1	1
2	2	1
3	2	2

Output	os	ot
1	2	1
2	2	1
3	3	2

Transition	Name
1	infection
2	recovery



1. Functorial generation of code



State	Name
1	S
2	I
3	R

Input	is	it
1	1	1
2	2	1
3	2	2

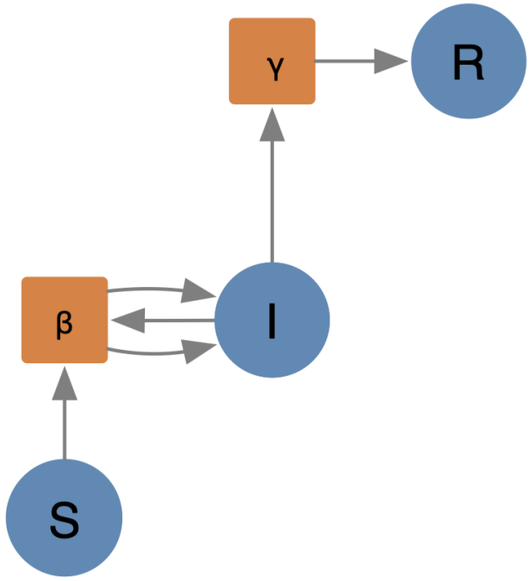
Output	os	ot
1	2	1
2	2	1
3	3	2

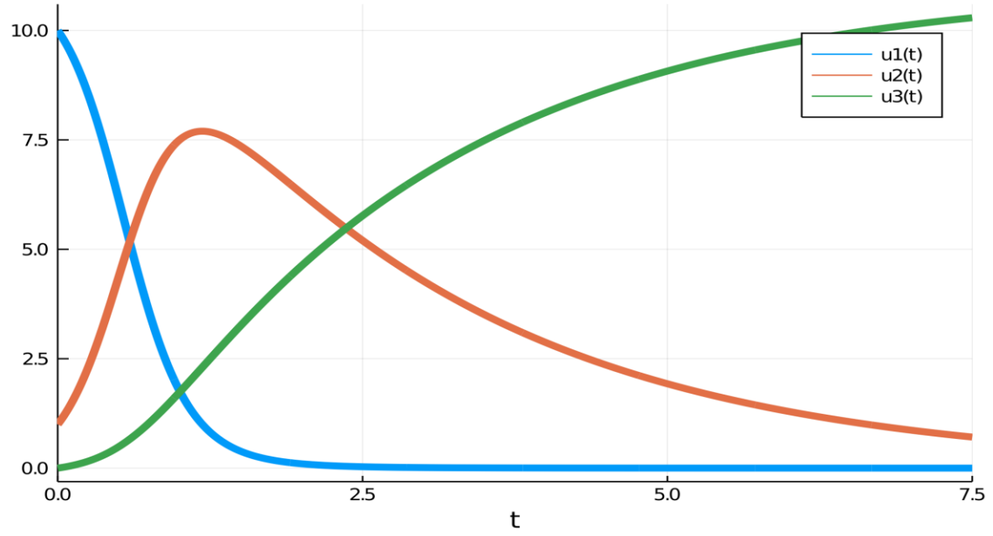
Transition	Name
1	infection
2	recovery

Specifying a reaction network as a Petri Net allows for automatically generating an ODE simulation.

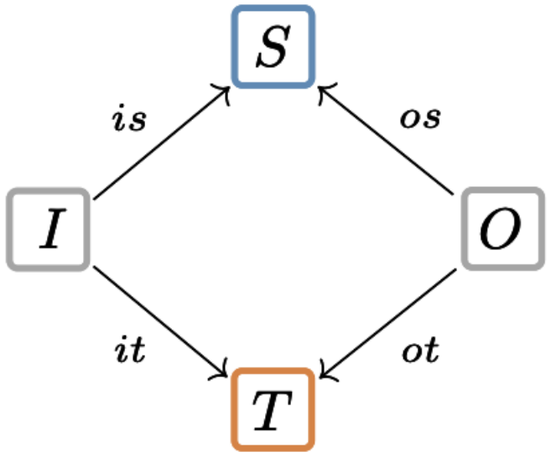
$$r_\beta = k_\beta \cdot [S] \cdot [I]$$

$$r_\gamma = k_\gamma \cdot [I]$$





2. Easily change semantics with different functor

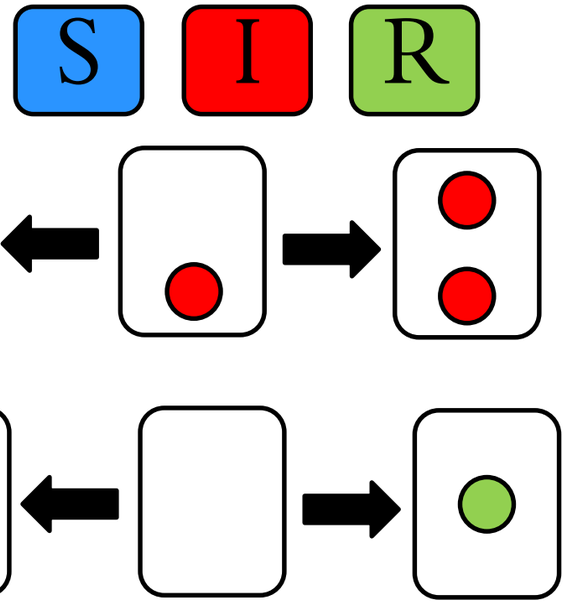


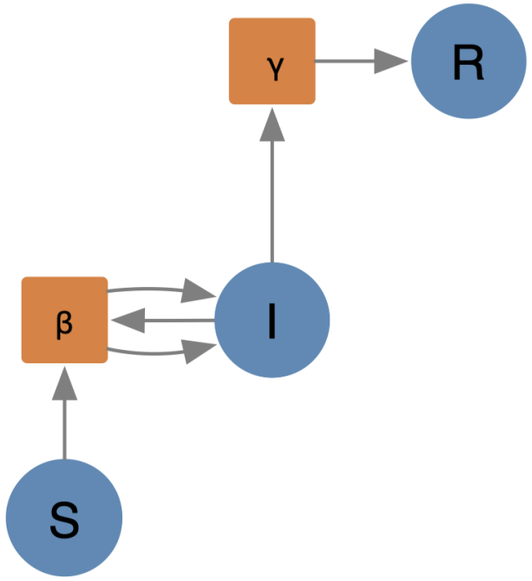
State	Name
1	S
2	I
3	R

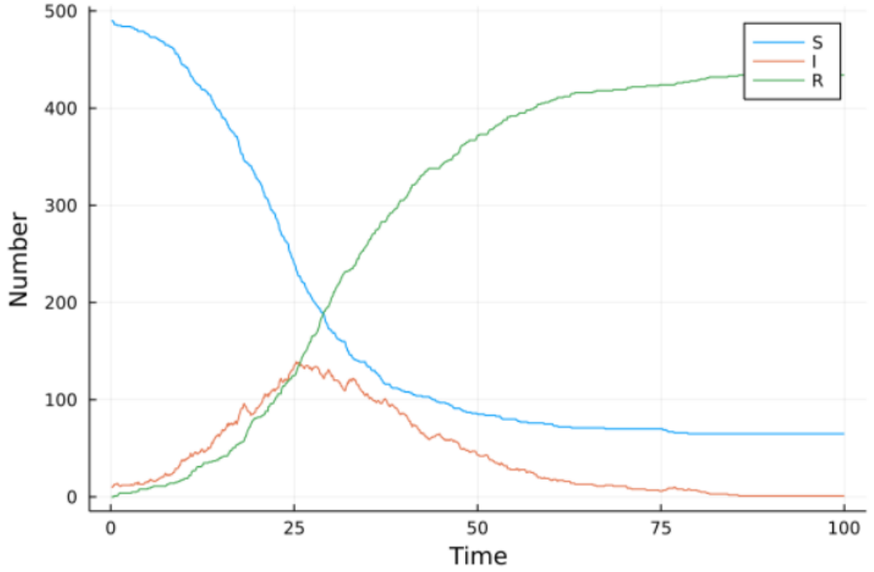
Input	is	it
1	1	1
2	2	1
3	2	2

Output	os	ot
1	2	1
2	2	1
3	3	2

Transition	Name
1	infection
2	recovery



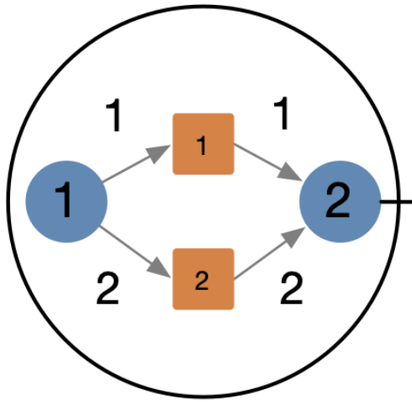


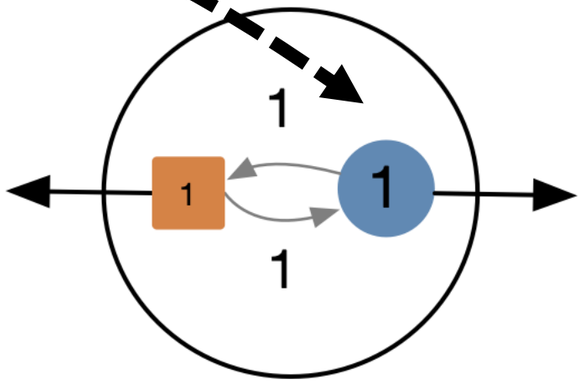


3. Build models compositionally (operad)

Database queries can be built hierarchically using a wiring diagram syntax. Raw SQL queries are not composable this way.

"Find all catalysts (*that are the product of two reactions*) and the reactions they catalyze"



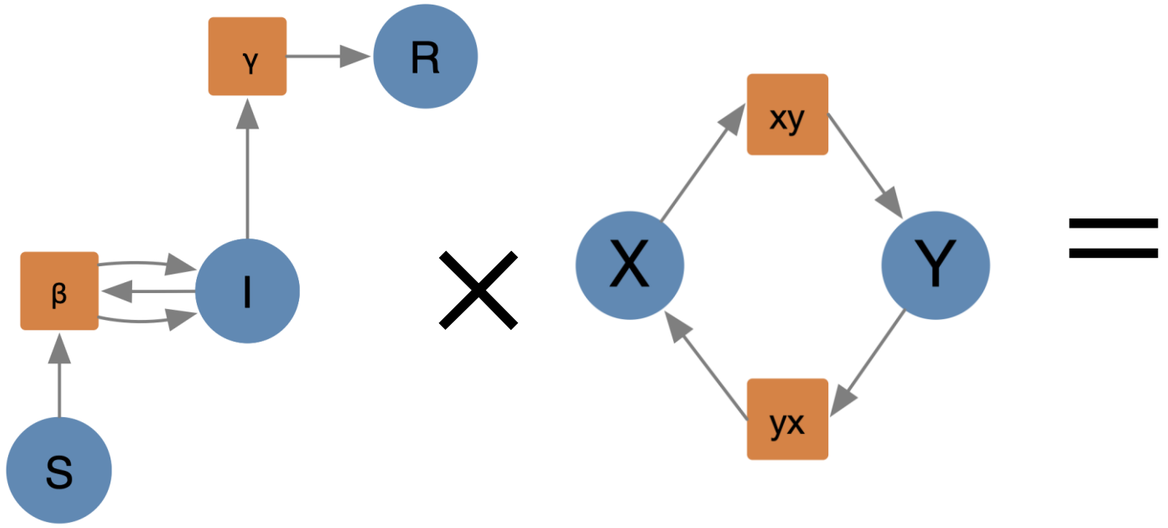



```
SELECT state2.id
FROM   S AS state1, S AS state2,
       T AS tran1,  T AS tran2,
       I AS in1,    I AS in2,
       O AS out1,   O AS out2
WHERE  in1.is  = state1.id,
       in1.it  = tran1.id
       out1.os = state2.id
       out1.ot = tran1.id
       …
```

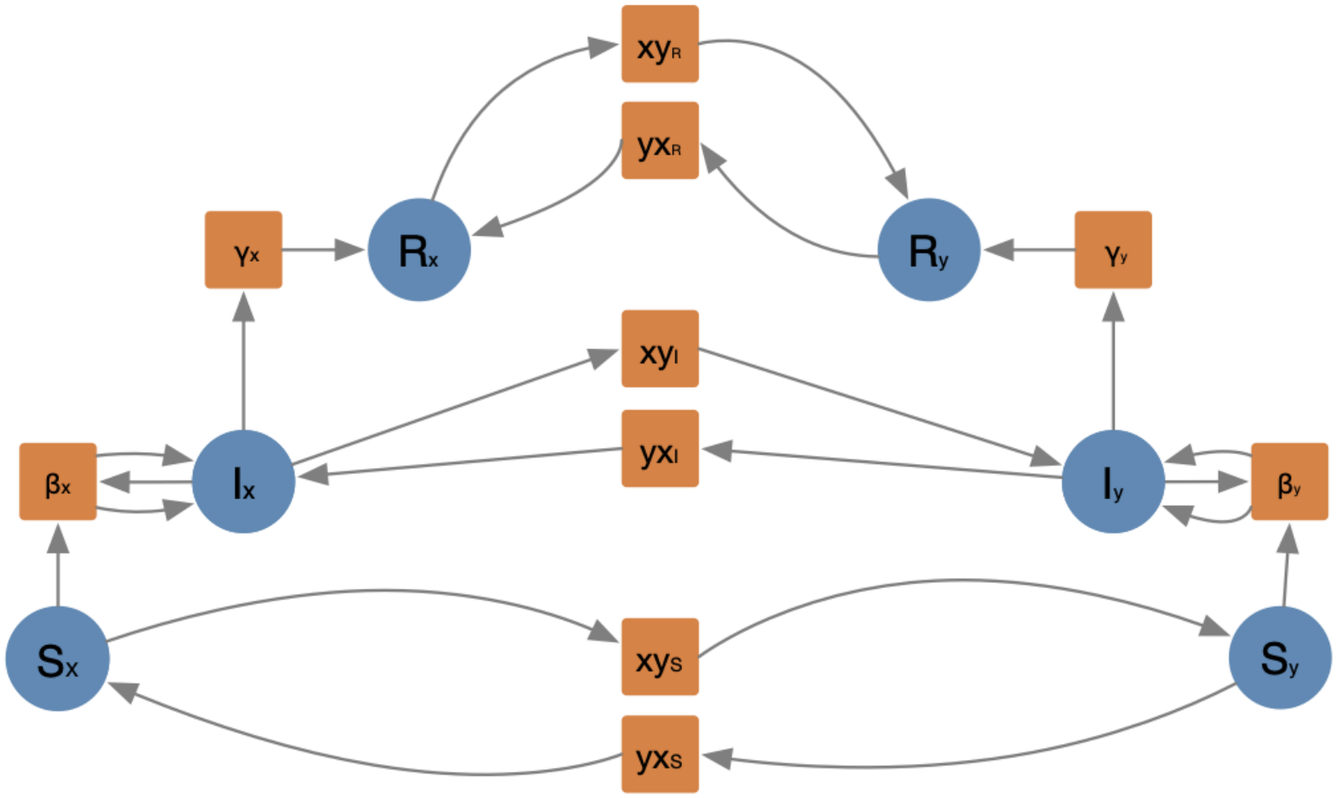
```
SELECT tran1.id, state1.id
FROM   S AS state1, T AS tran1,
       I AS in1,    O AS out1
WHERE  in1.is  = state1.id,
       in1.it  = tran1.id
       out1.os = state1.id
       out1.ot = tran1.id
```

4. Build models compositionally (limits)

"SIR model" "Two-city model"



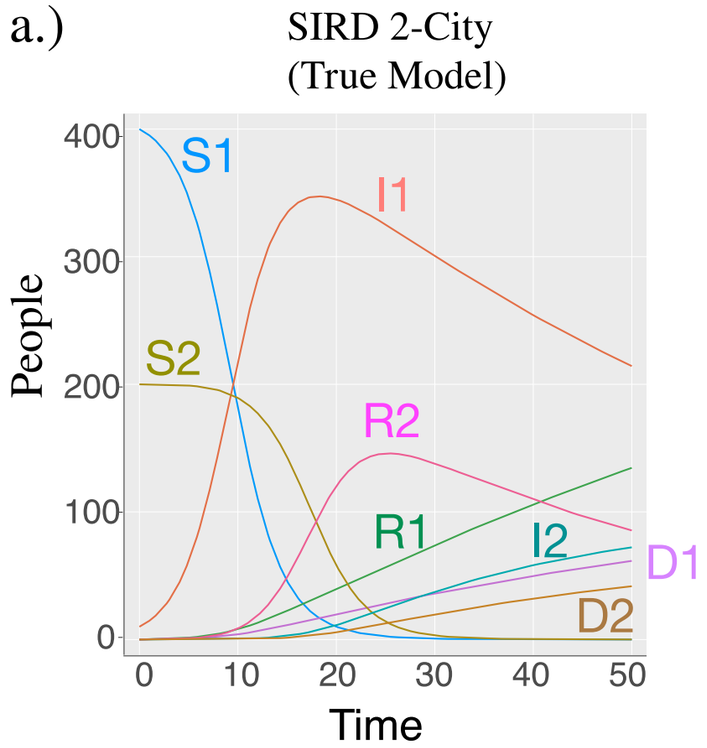
"Two-city SIR model"



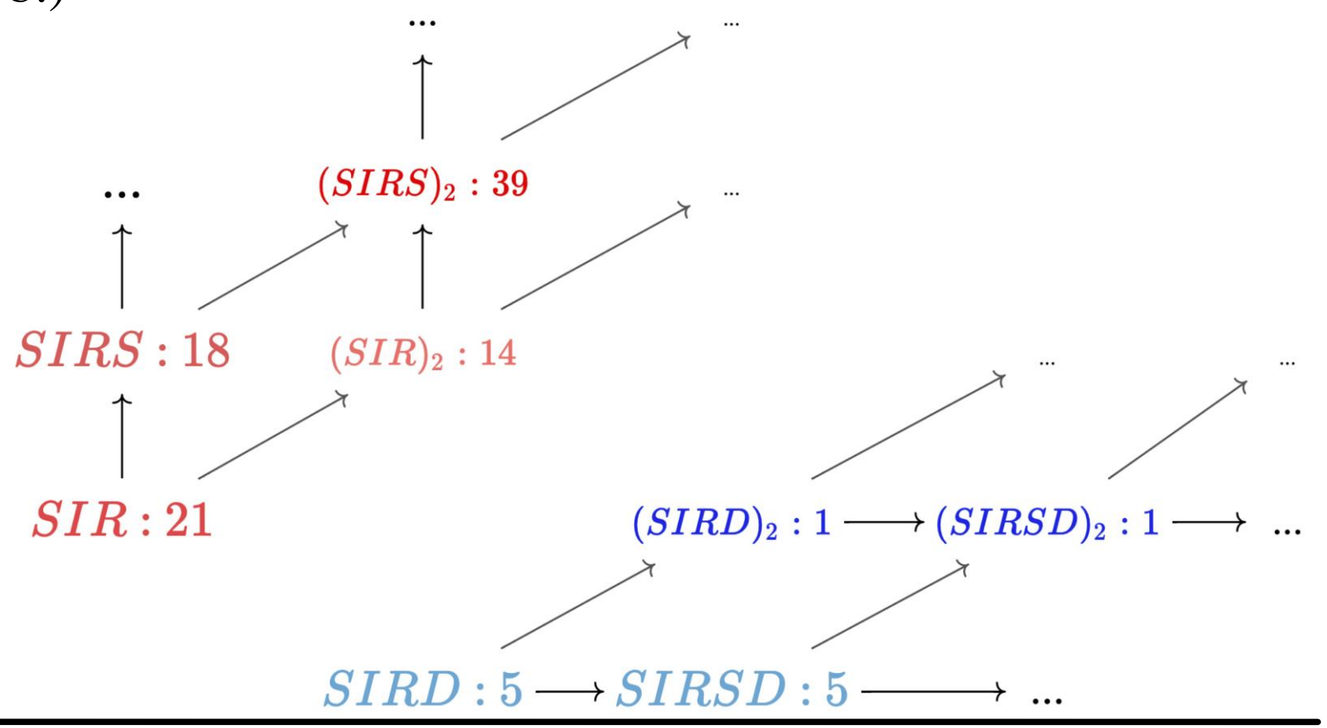
High-level operations on Petri nets like products do the 'right' thing' for reaction networks, unlike for symbolic syntax or raw ODEs.

5. Automated Model Selection

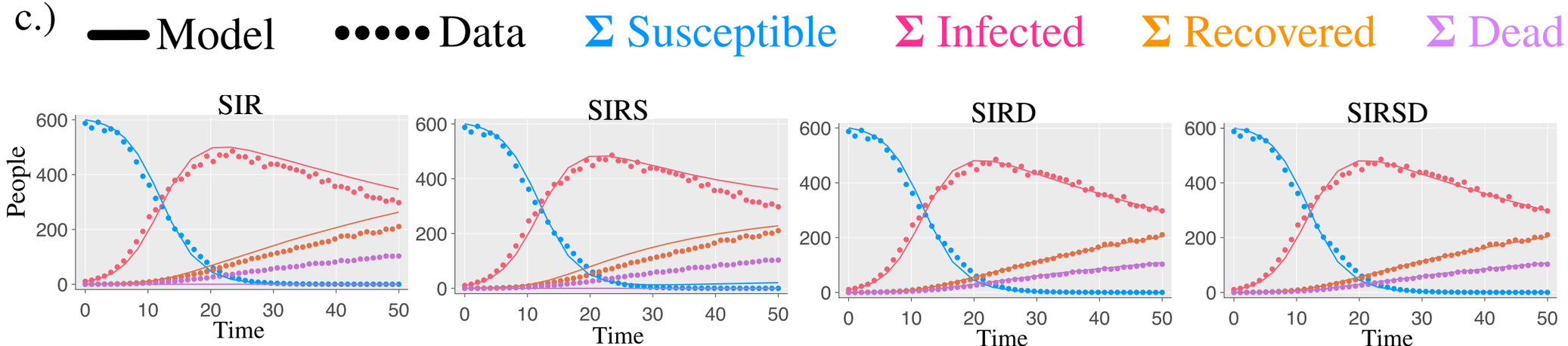
a.)



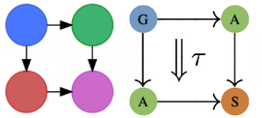
b.)



c.)



Outline



UF

Introduction

- Why represent scientific models combinatorially?

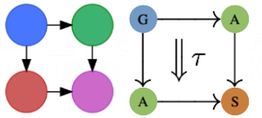
Model space exploration

- The category of diagrams as a category of model spaces
- Example limits and colimits
- Composition recipes
- Limits and colimits: implemenation

Model Selection

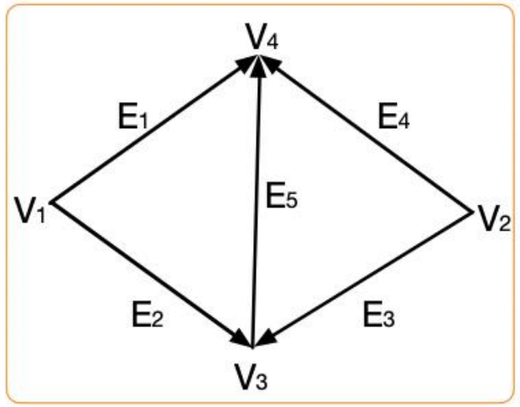
- Best fit chemical reaction network example

The category of diagrams

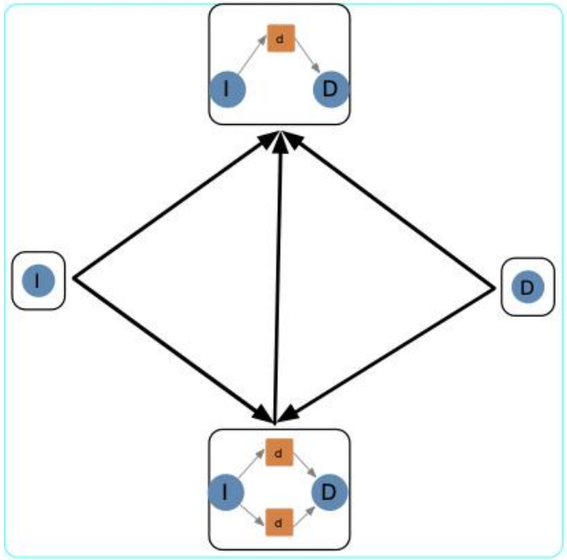


UF

A particular diagram in Petri



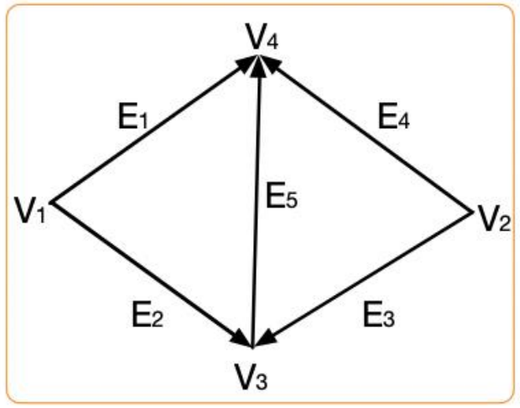
Shape



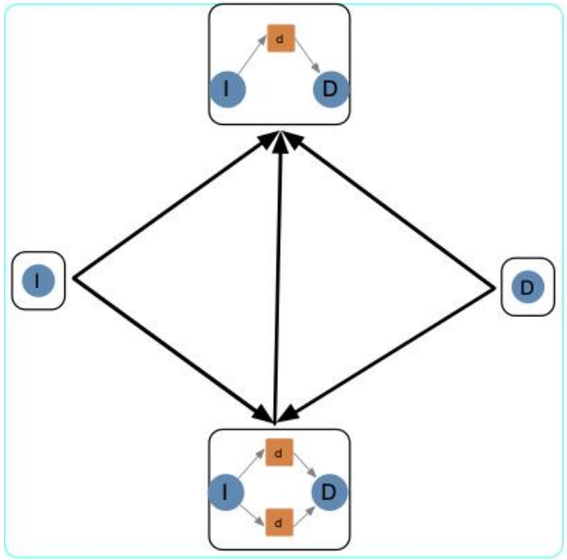
Diagram

The category of diagrams: a lax slice category

A particular diagram in Petri



Shape

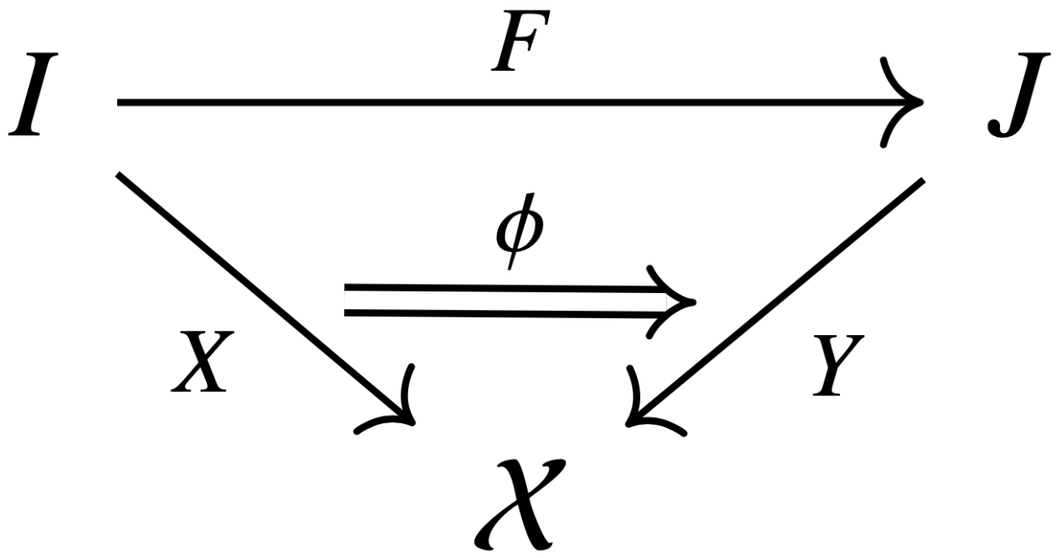


Diagram

$$(F, \phi): (I, X) \to (J, Y)$$

Shape map: F

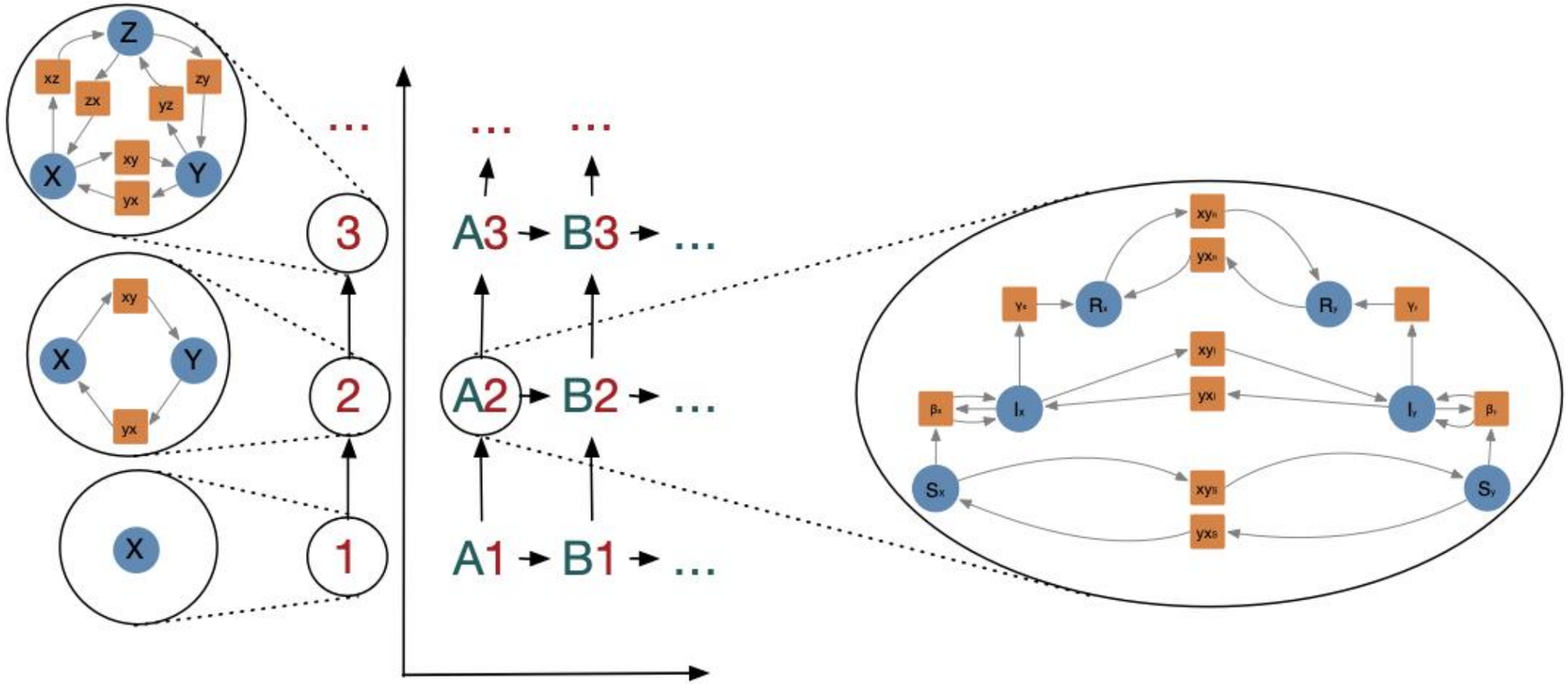
Diagram map: ϕ



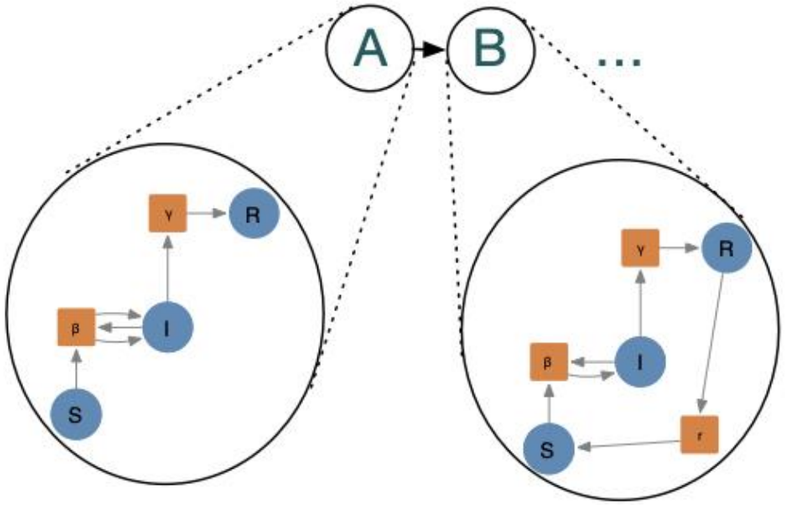
We want pushouts and pullbacks of diagrams

Model space product

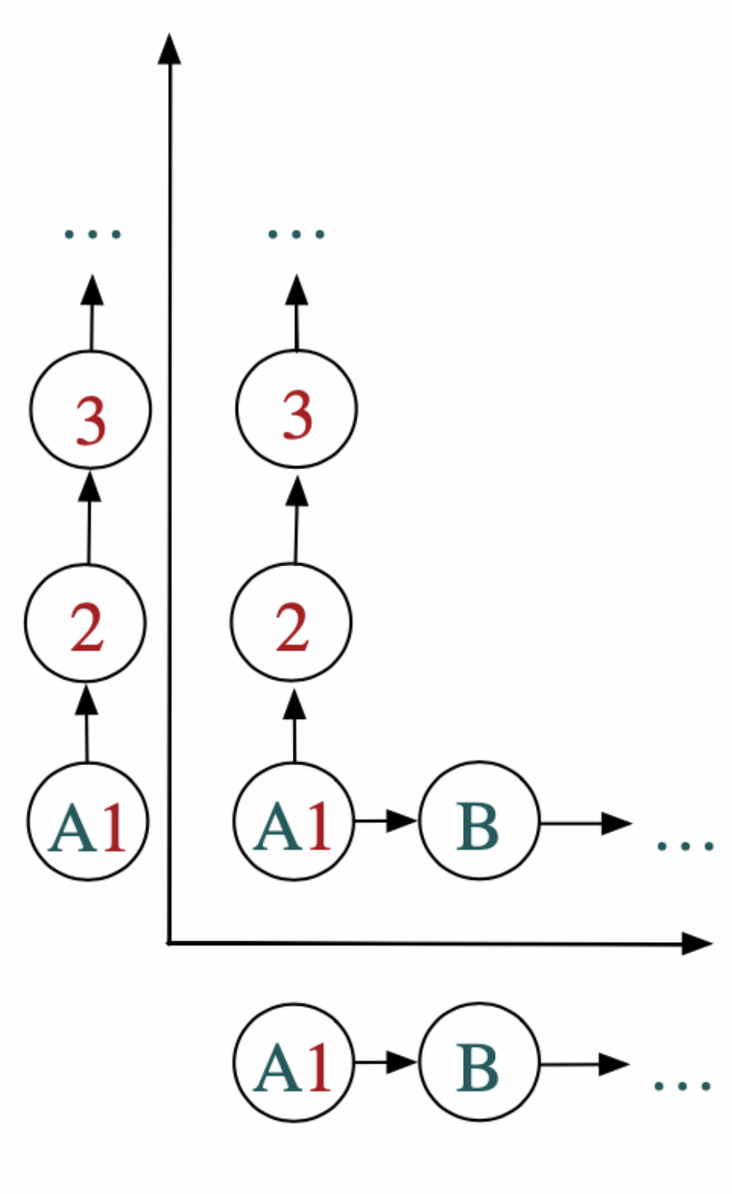
"City dimension"



"Disease dimension"

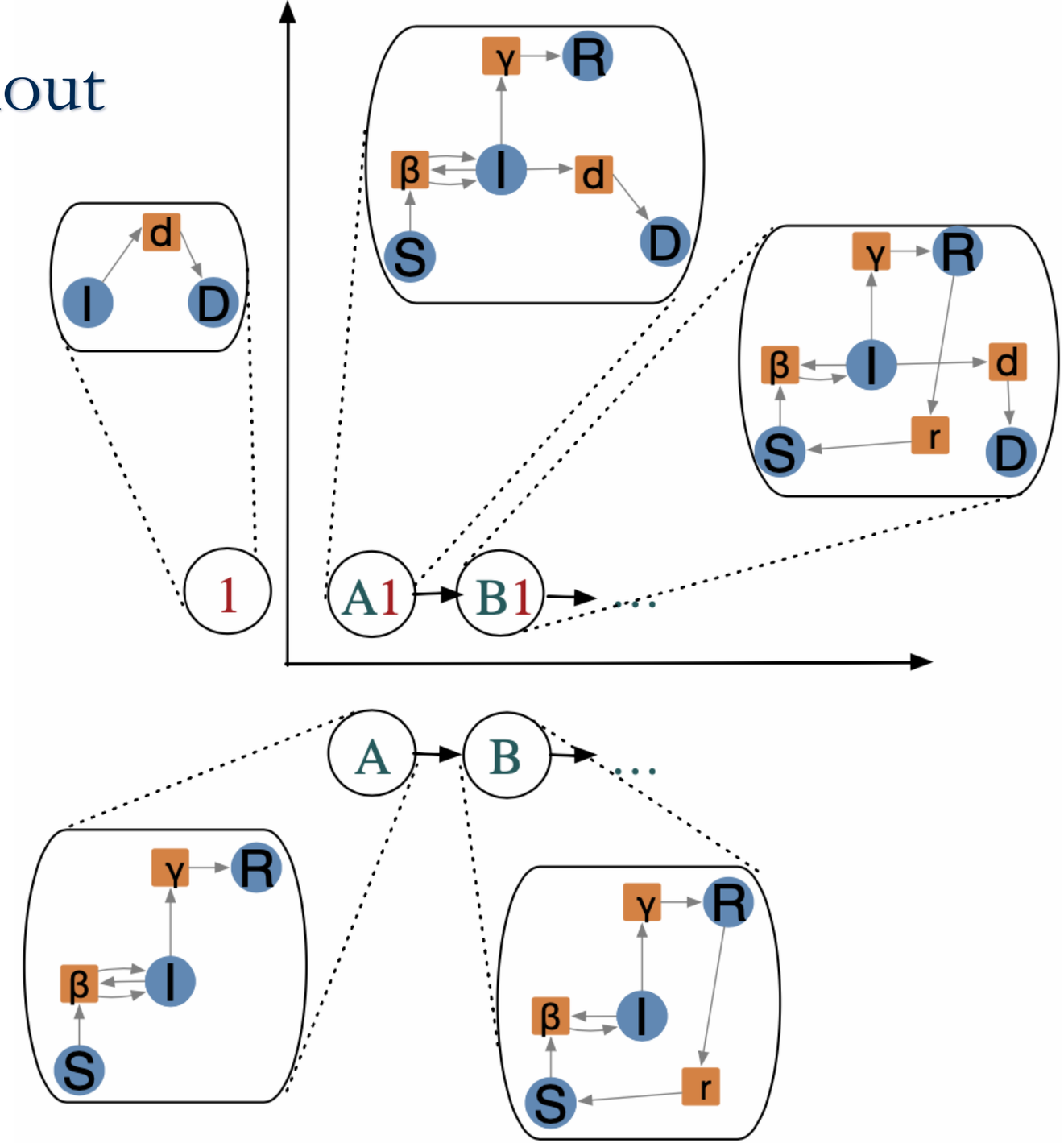


Model space pushout

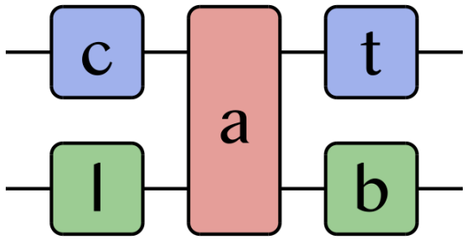


Model space pushout

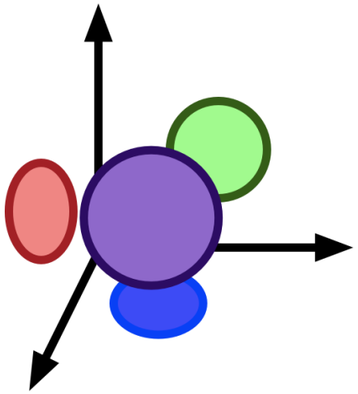
"Glue"



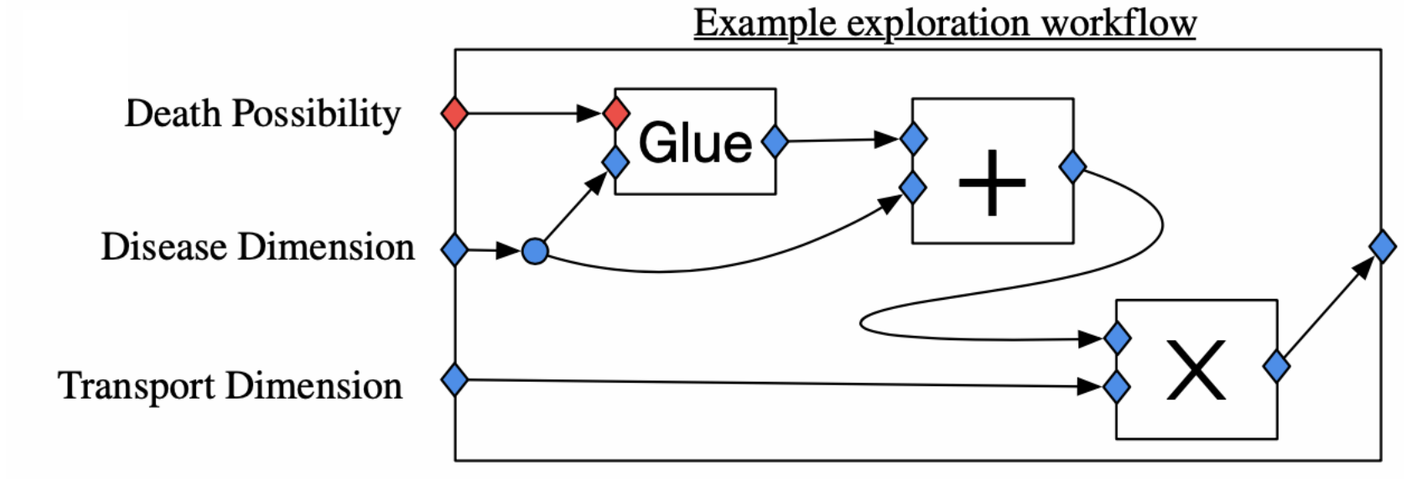
Composition

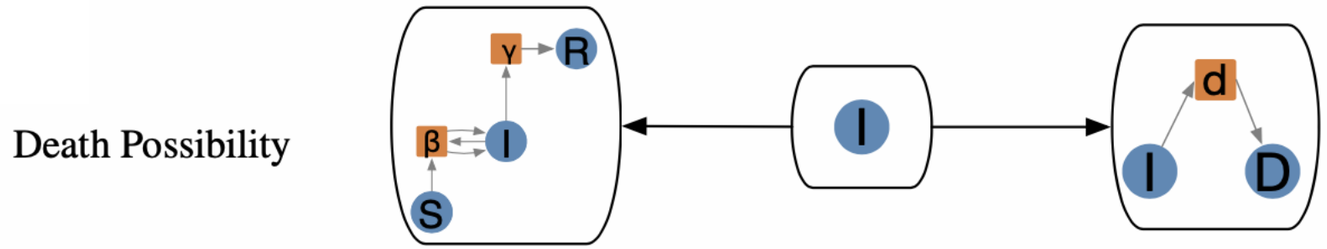


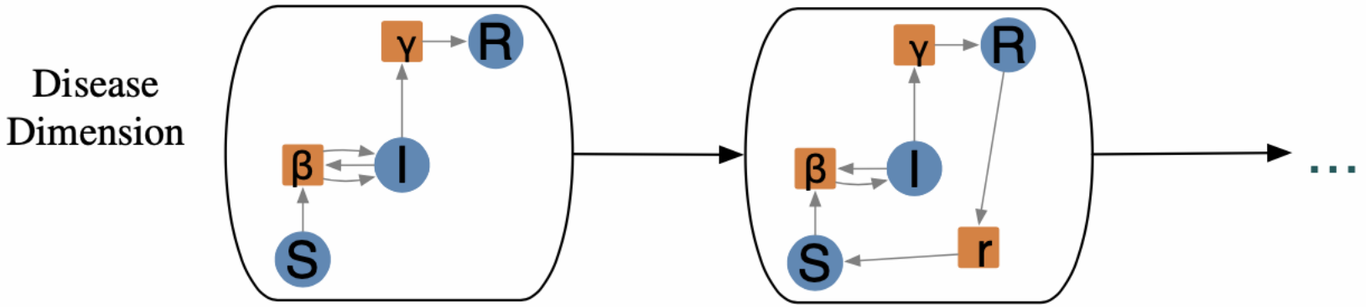
Catlab.jl

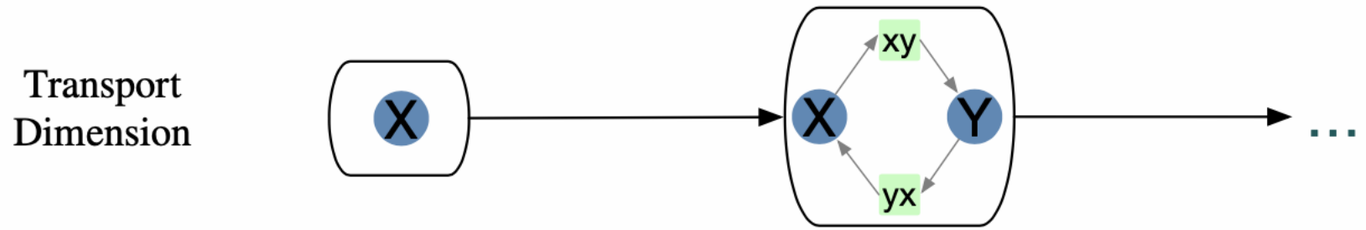


ModelExploration.jl

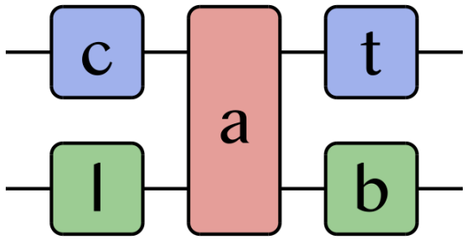




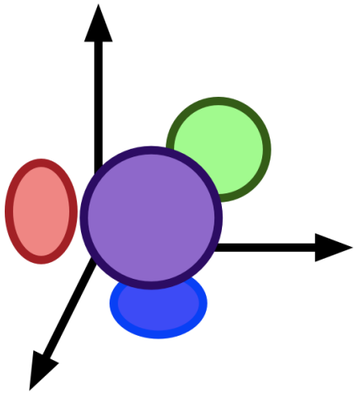




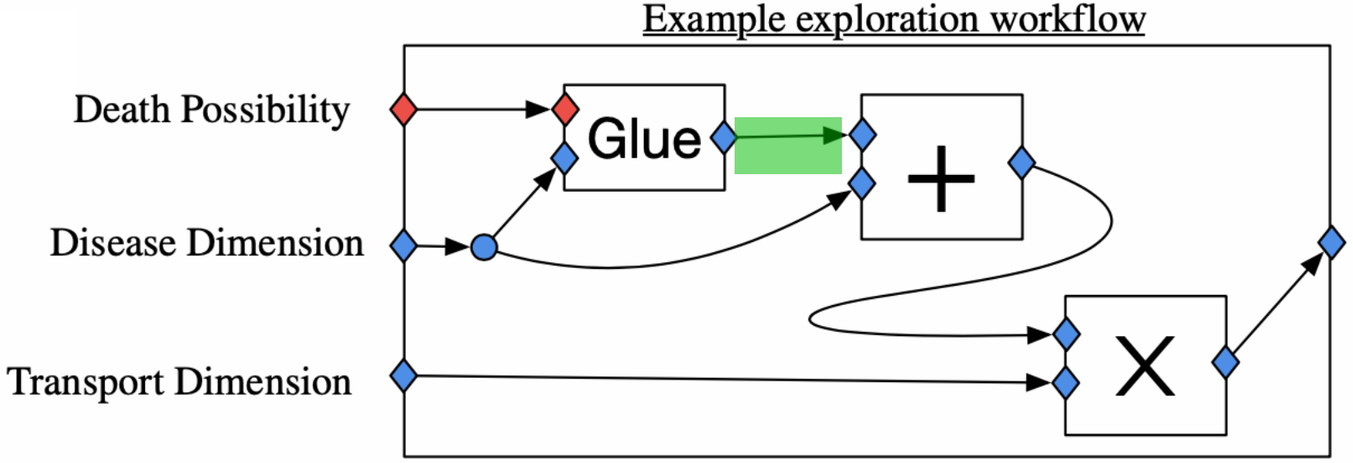
Composition result

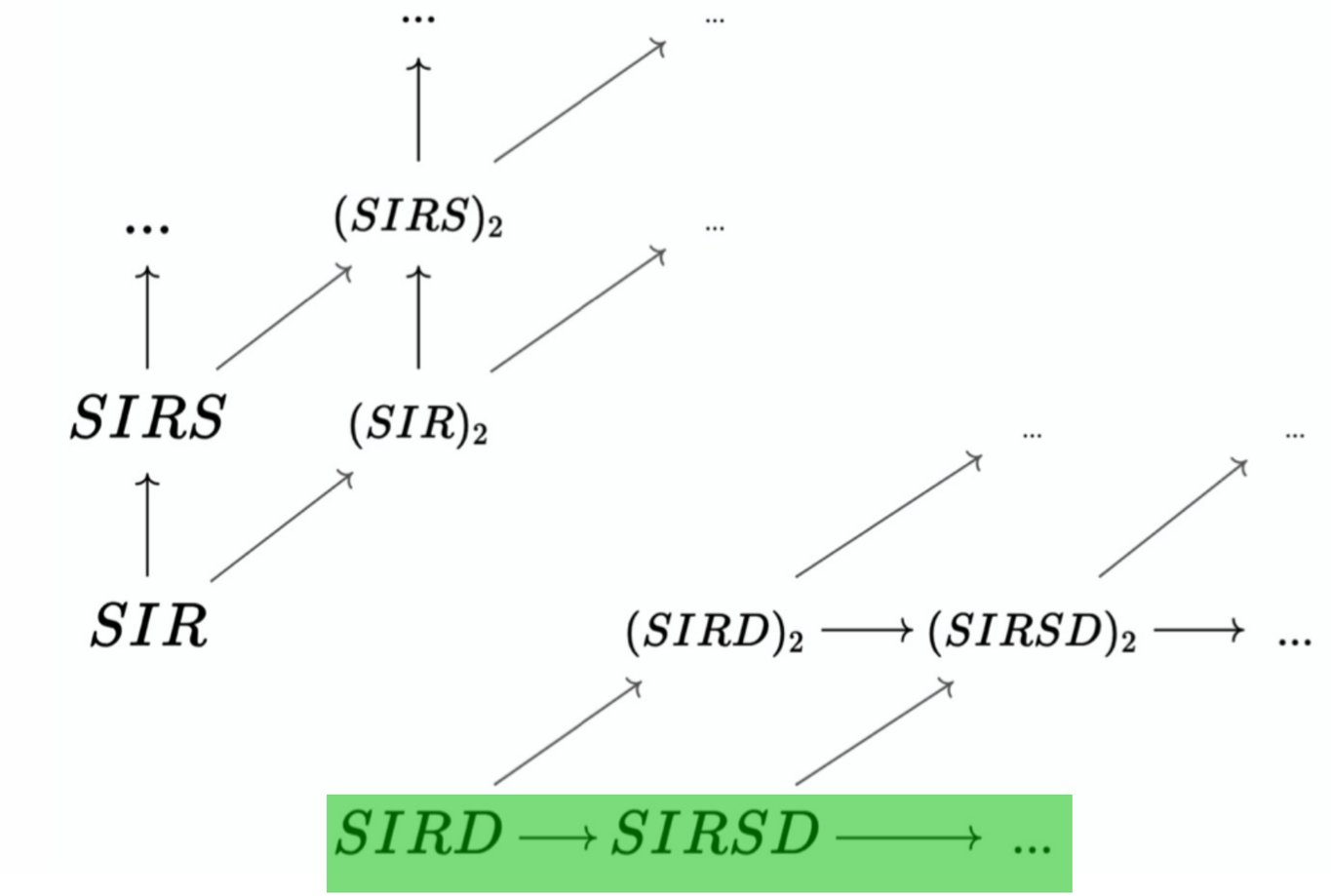


Catlab.jl

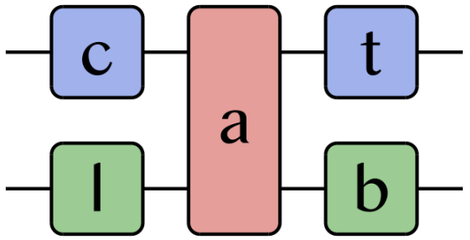


ModelExploration.jl

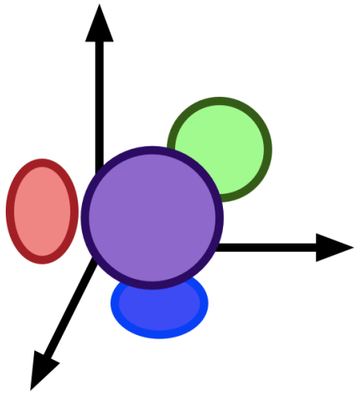




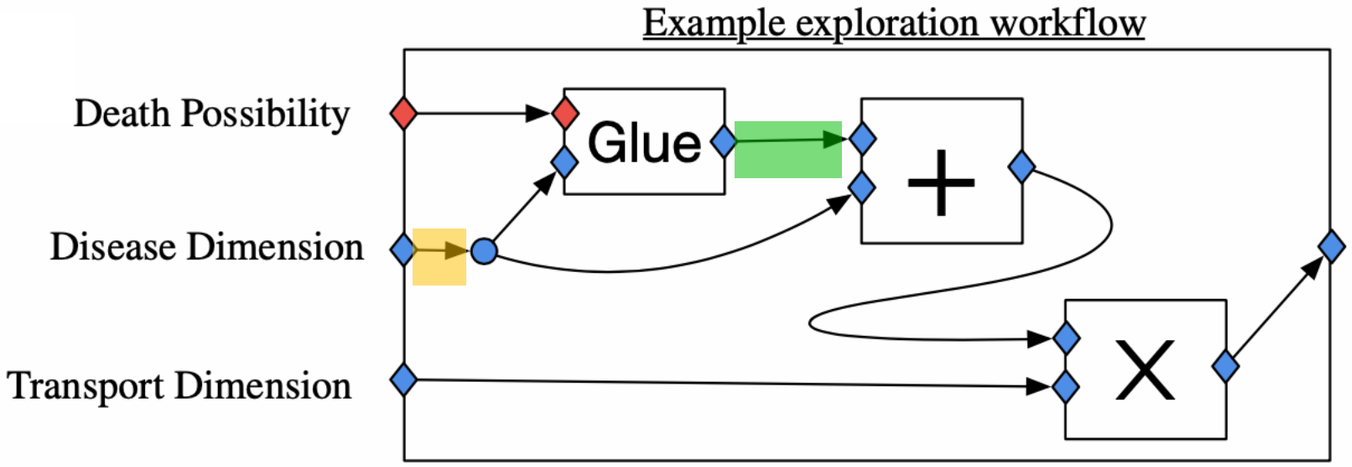
Composition result

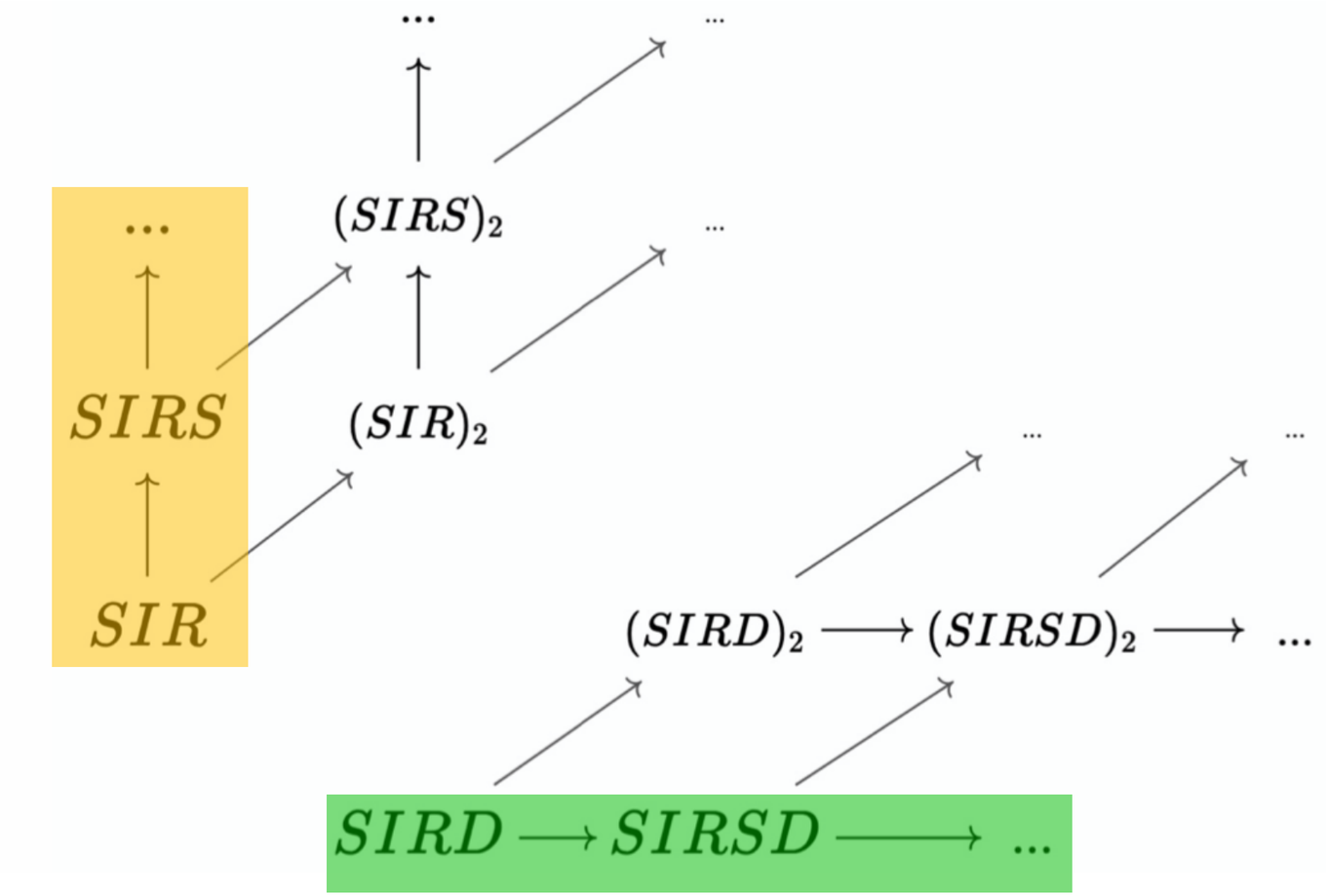


Catlab.jl

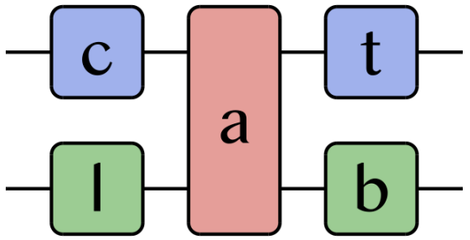


ModelExploration.jl

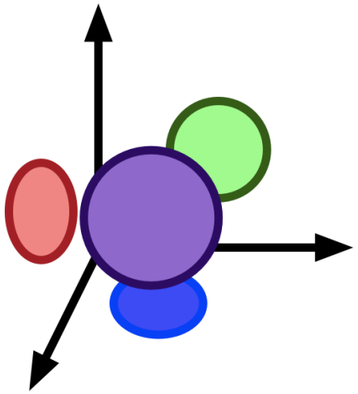




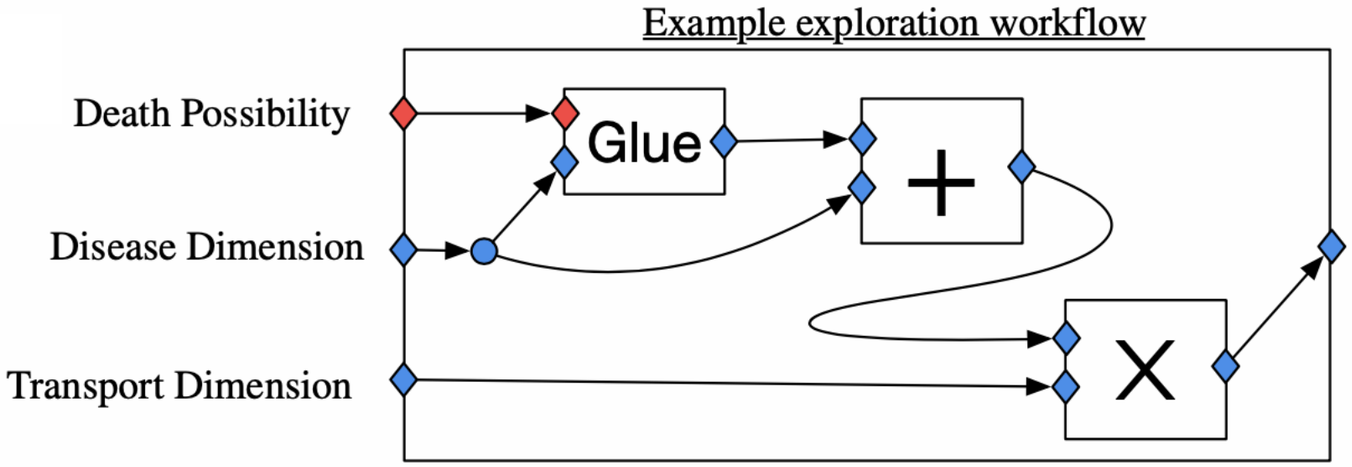
Composition result

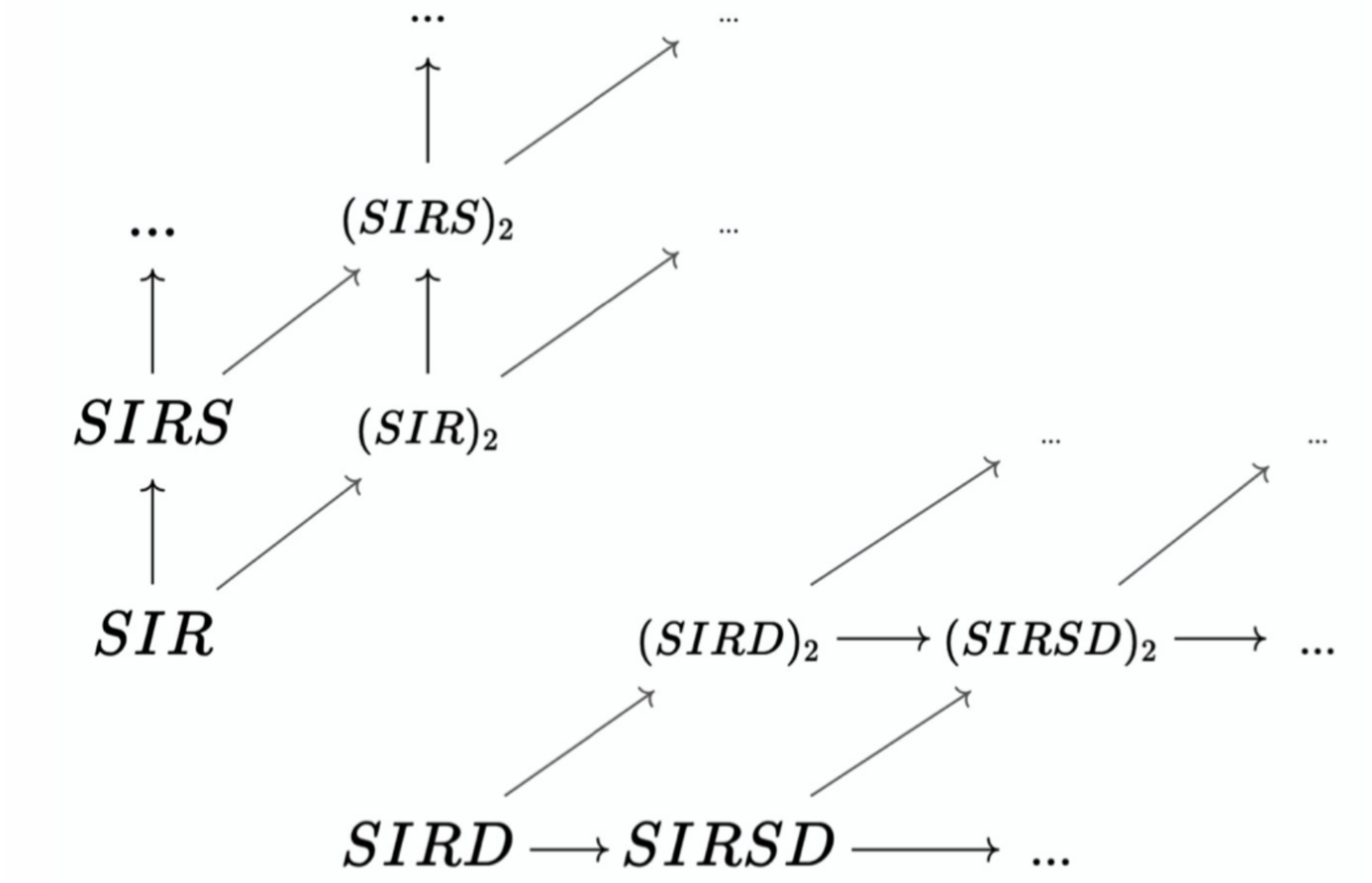


Catlab.jl



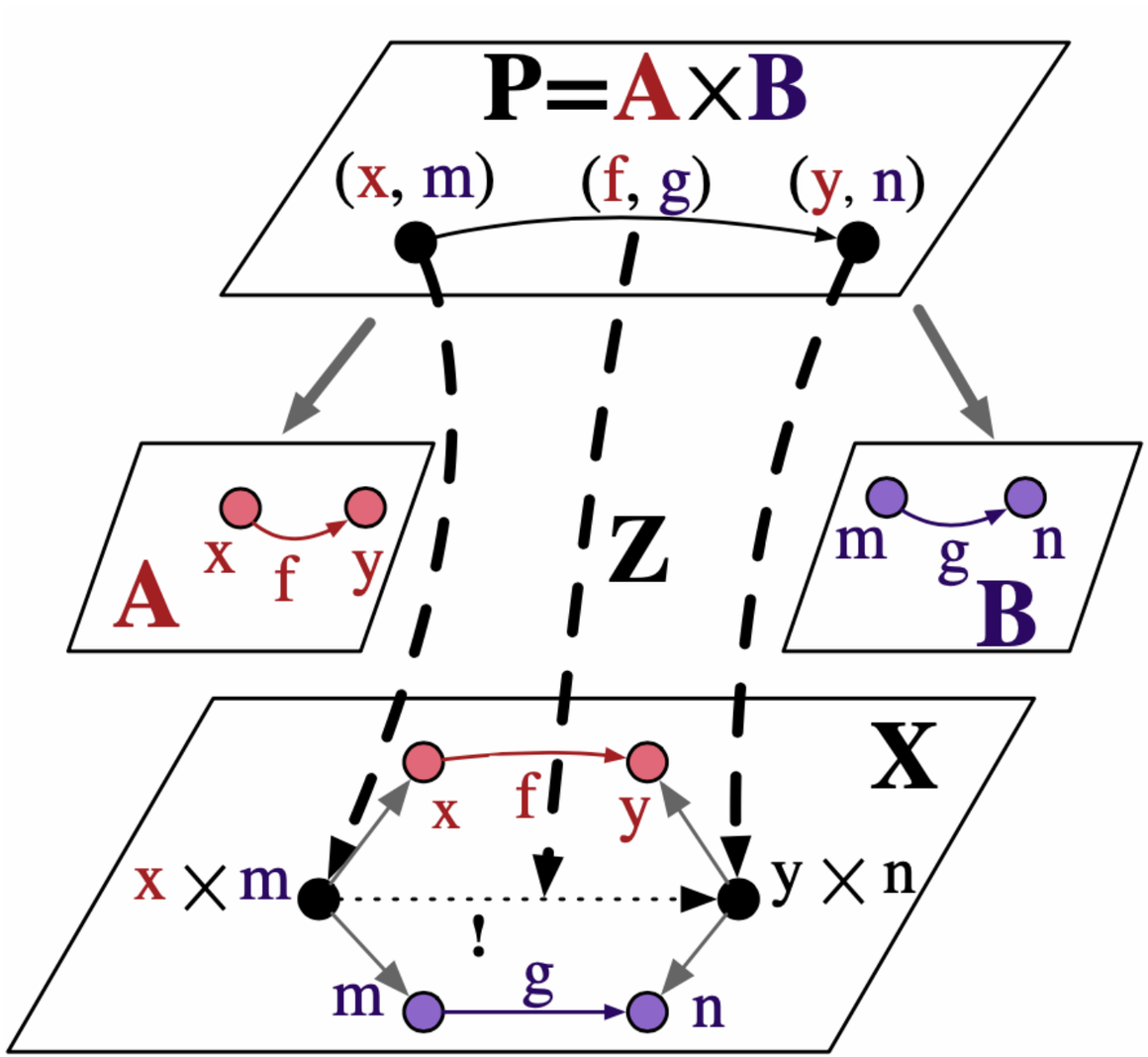
ModelExploration.jl





Limits - implementation

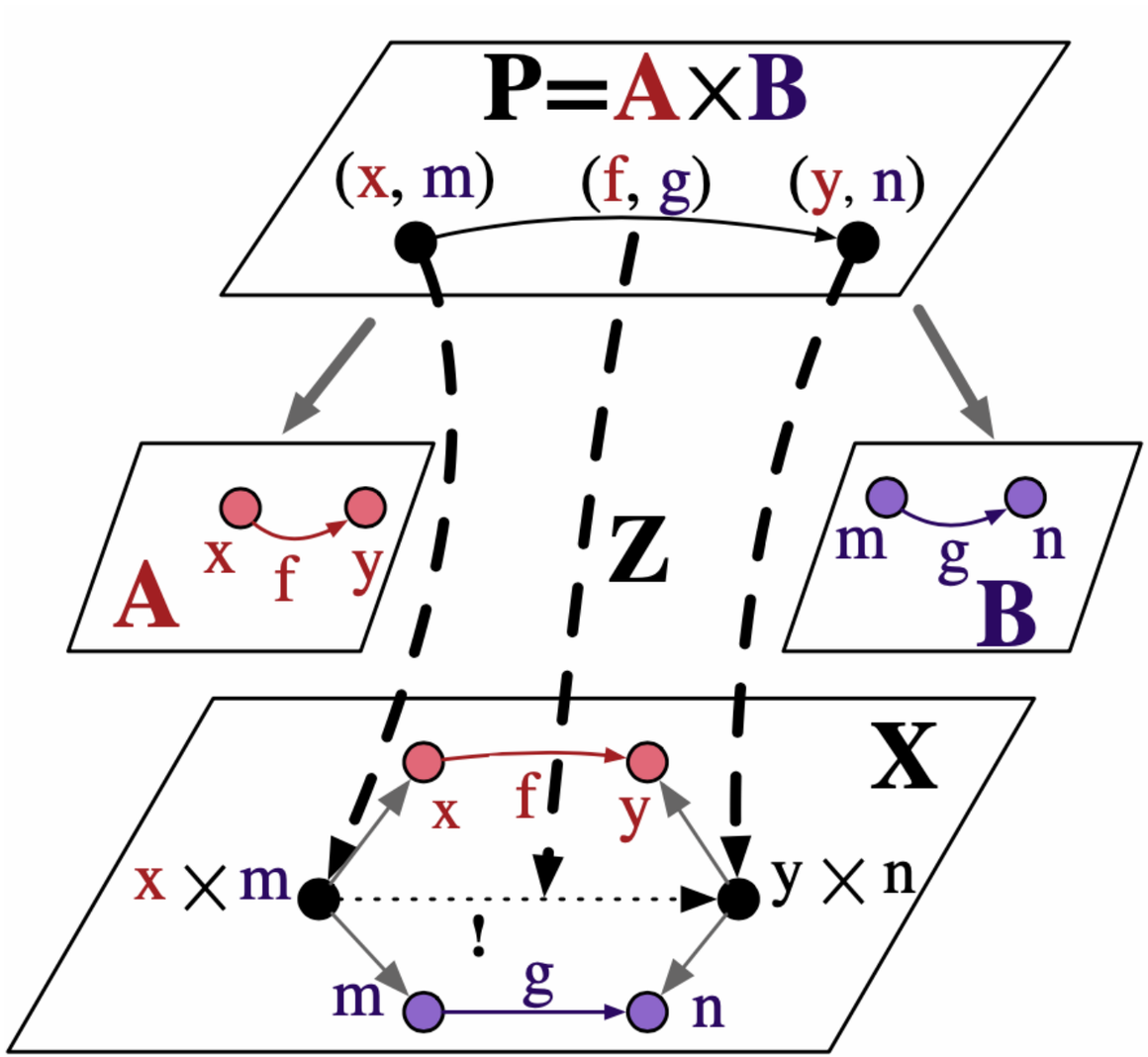
Product



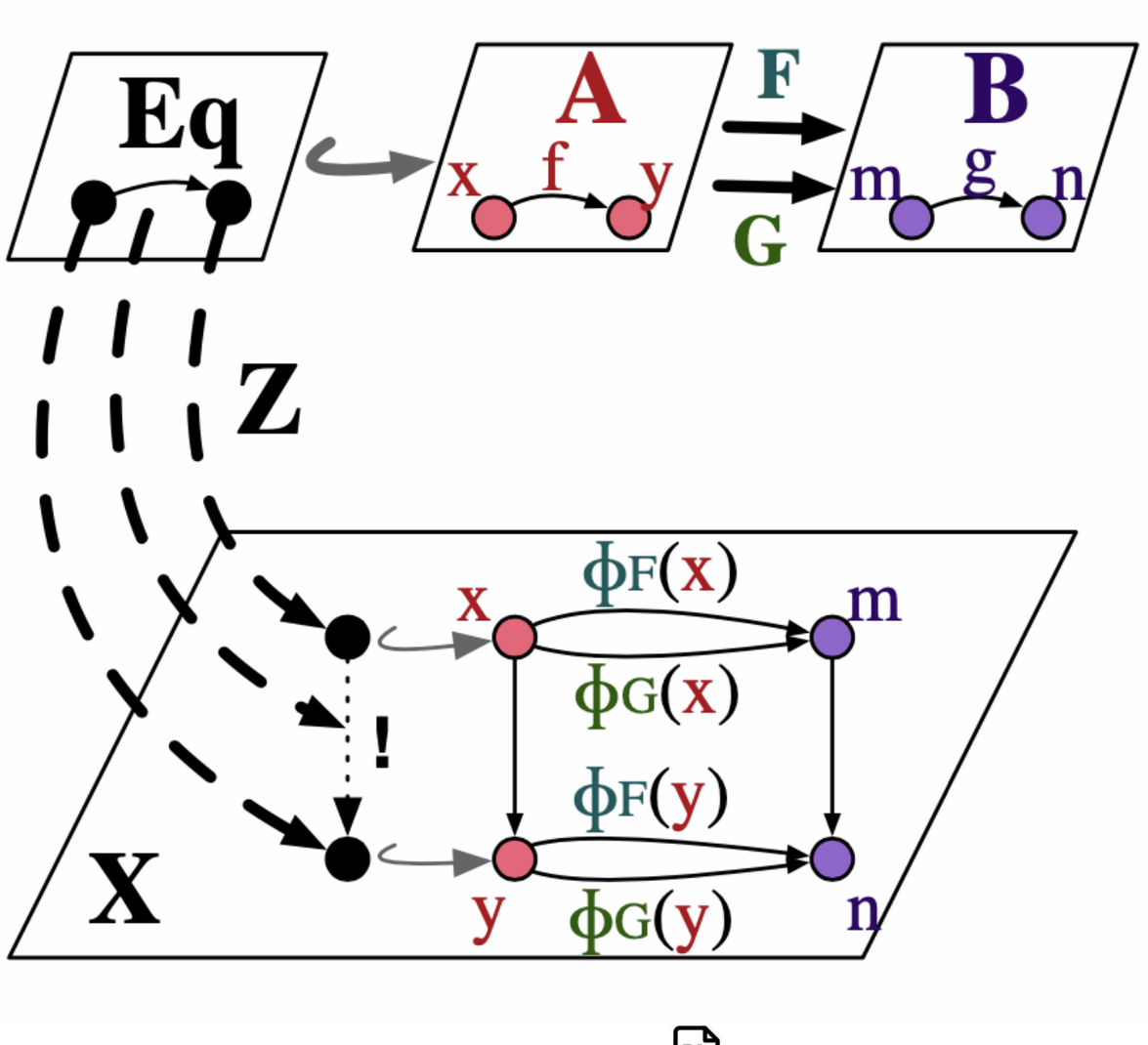
Peschke and Tholen (2020)

Limits - implementation

Product



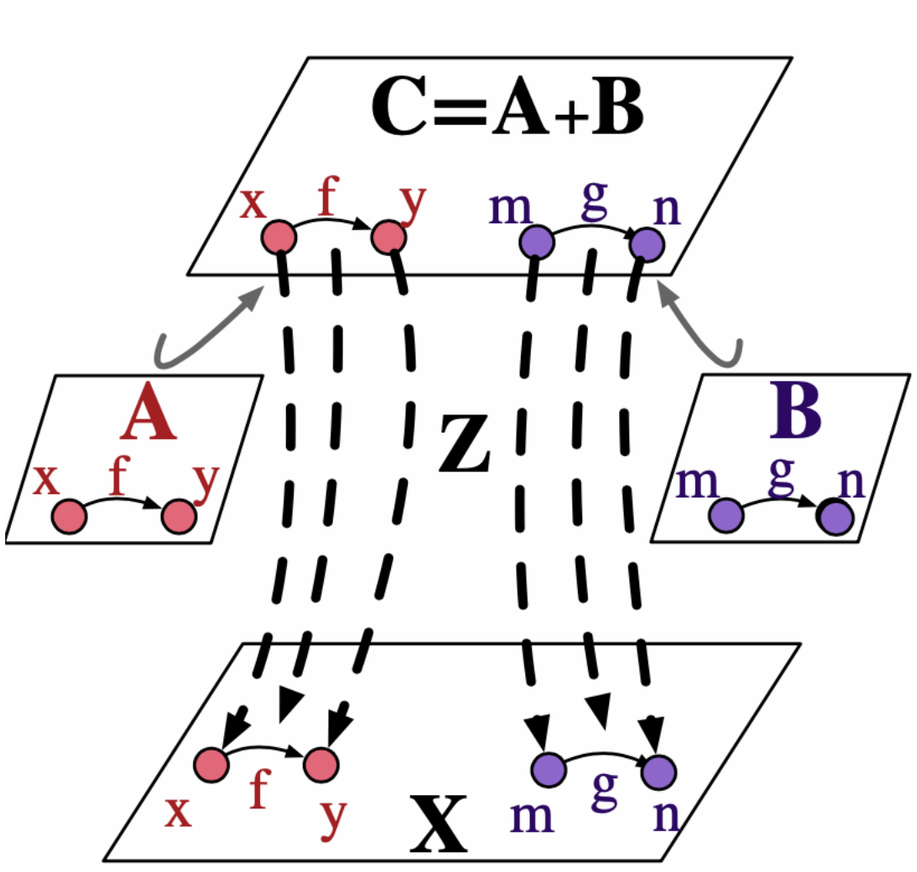
Equalizer



Peschke and Tholen (2020)

Colimits - implementation

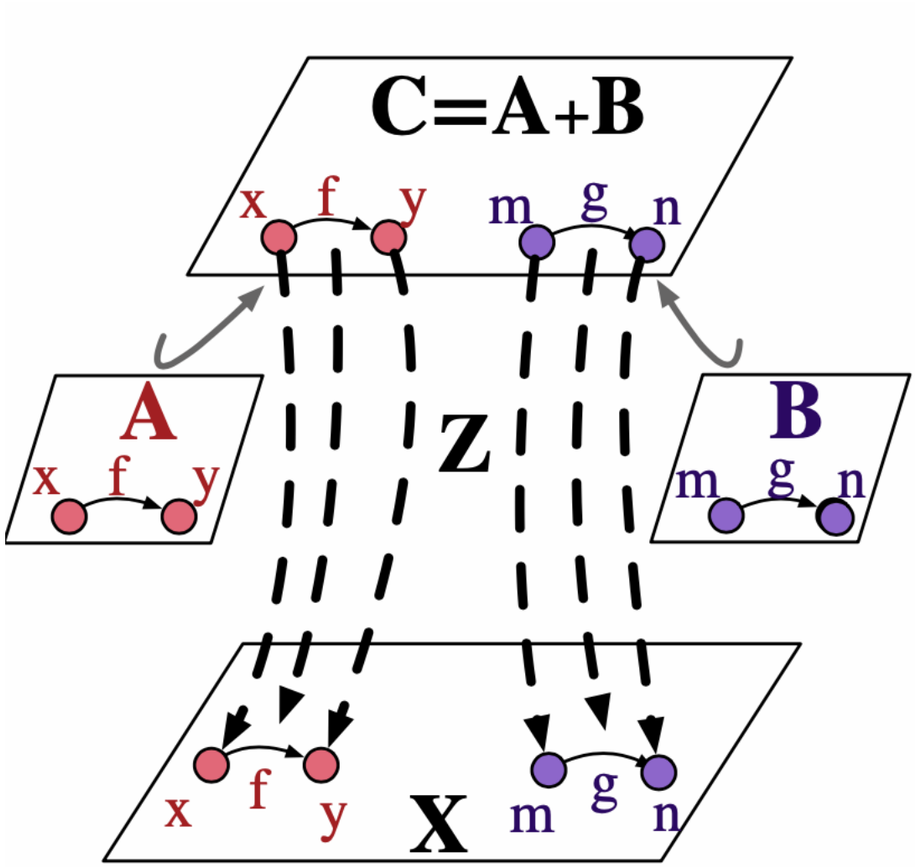
Coproduct



Peschke and Tholen (2020)

Colimits - implementation

Coproduct

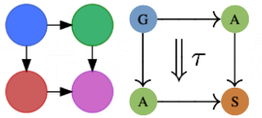


Coequalizer

$$
\begin{array}{ccc}
(A, X) & \xrightarrow{(FH,\ \lambda)} & (C,\ Lan_{FH}X) \\
{\scriptstyle (F,\ \phi_i)}\downarrow & & \downarrow{\scriptstyle (id_C,\ \alpha_i)} \\
(B, Y) & \xrightarrow{(H,\ \kappa)} & (C,\ Lan_H Y) \\
 & {\scriptstyle (H,\ \kappa\gamma)}\searrow & \downarrow{\scriptstyle (id_C,\ \gamma)} \\
 & & (C,\ Z)
\end{array}
$$

Peschke and Tholen (2020)

Outline



UF

Introduction

- Why represent scientific models combinatorially?

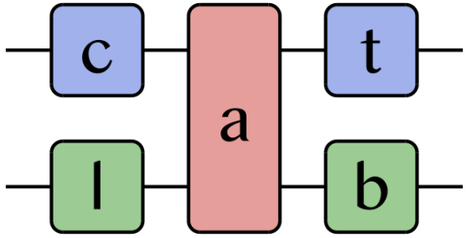
Model space exploration

- The category of diagrams as a category of model spaces
- Example limits and colimits
- Composition recipes
- Limits and colimits: implemenation

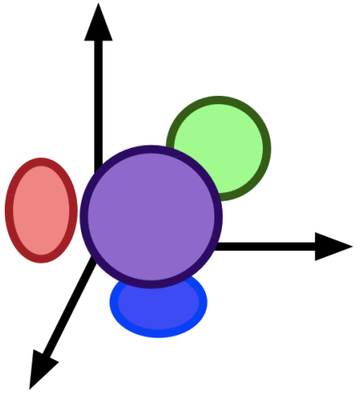
Model Selection

- Best fit chemical reaction network example

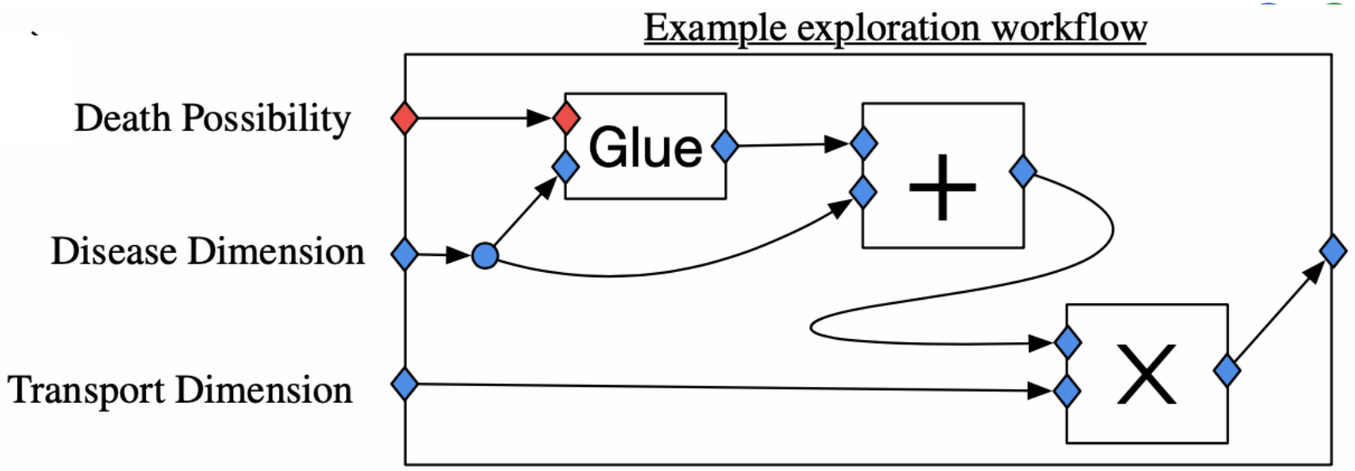
Composition result

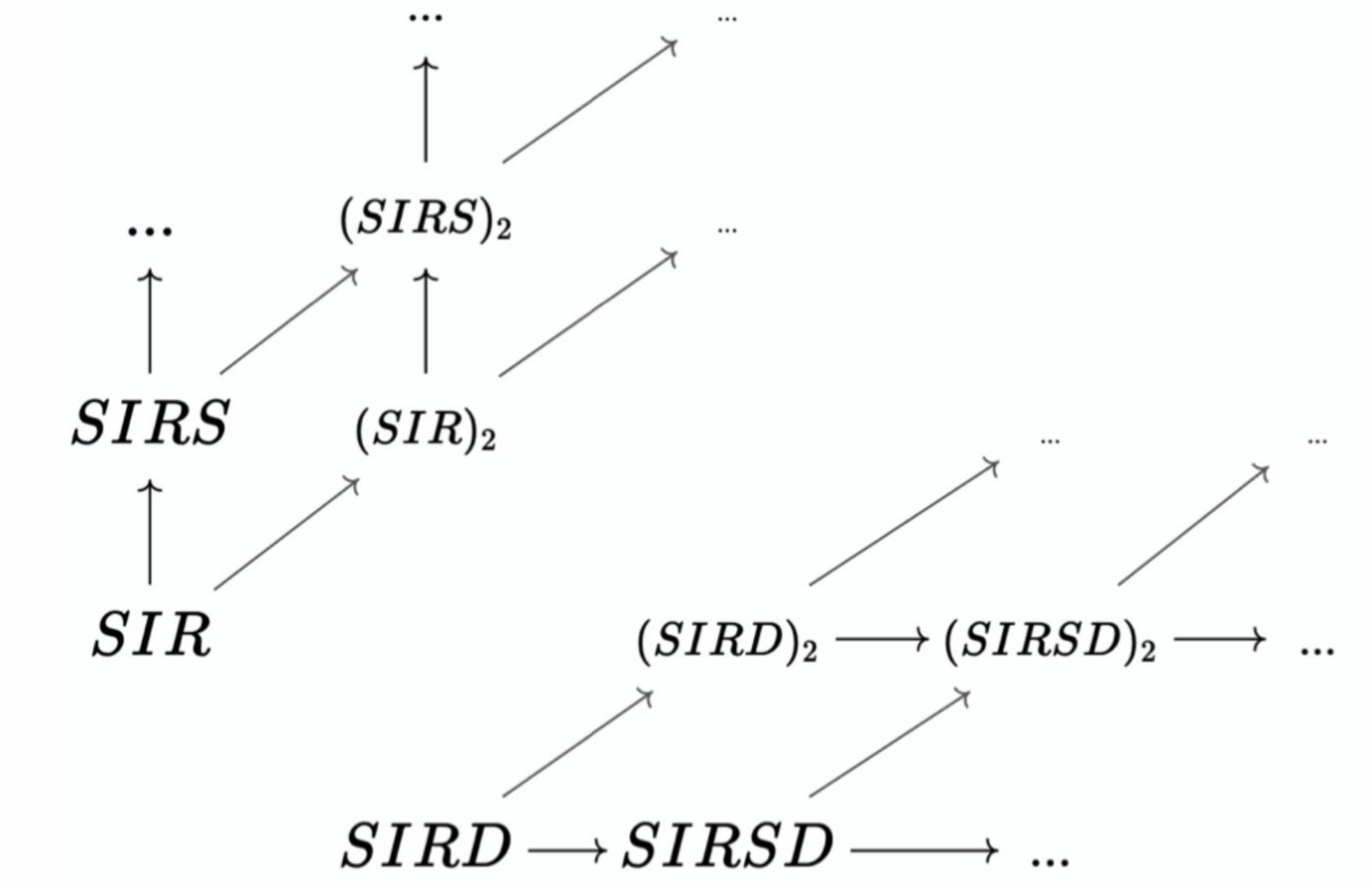


Catlab.jl



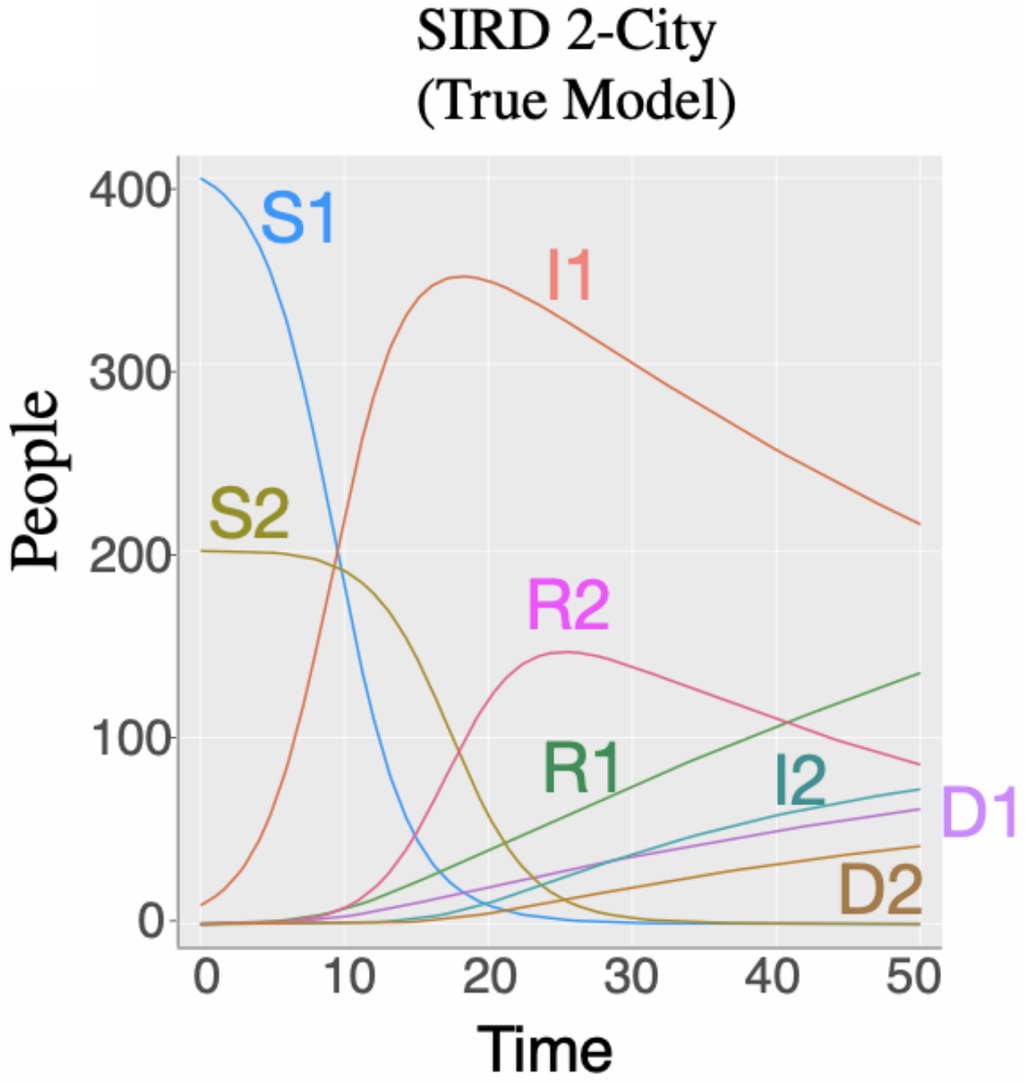
ModelExploration.jl





One particular model selection strategy

$$Loss(y, \hat{y}, \vec{p}) = \sum_{t \in T} \sum_{s \in \{S,I,R\}} (y_s(t) - \hat{y}_s(\hat{p},\ t))^2$$



Model selection results

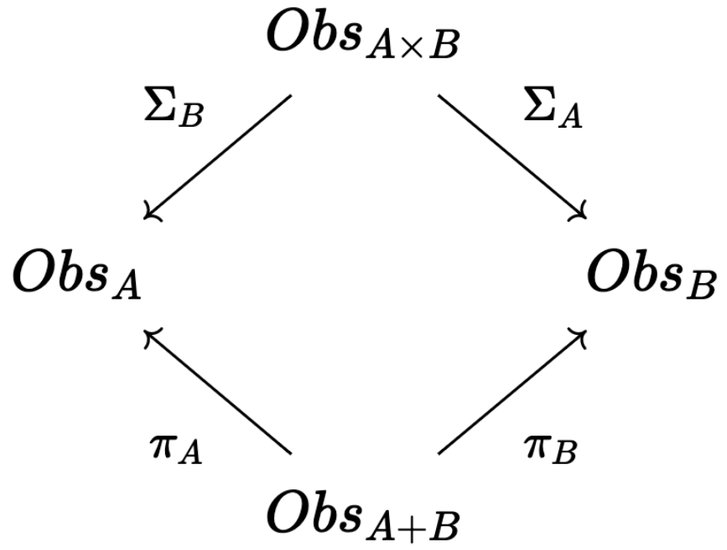
$$Param_{A\times B}$$

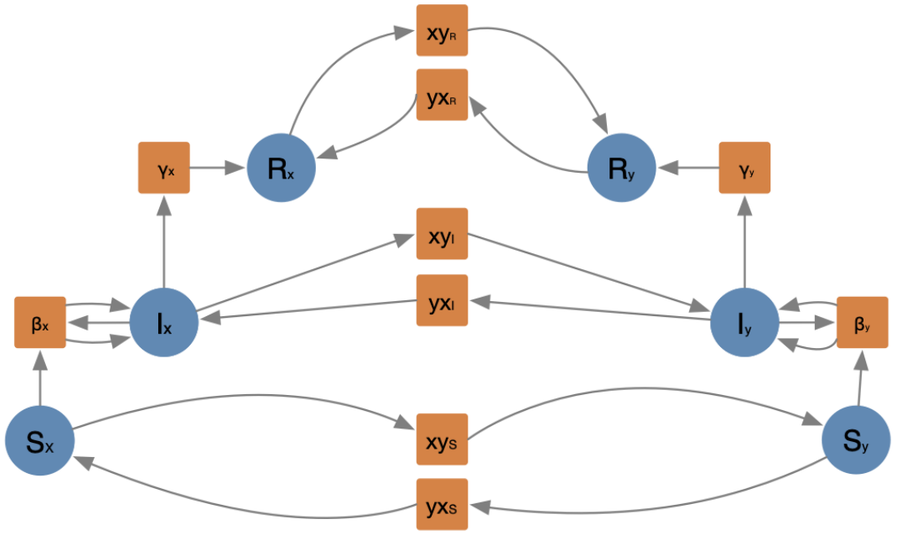
$$\uparrow \ comm.\ monoid$$

$$Param_A \times Param_B$$

$$\downarrow \ concat$$

$$Param_{A+B}$$





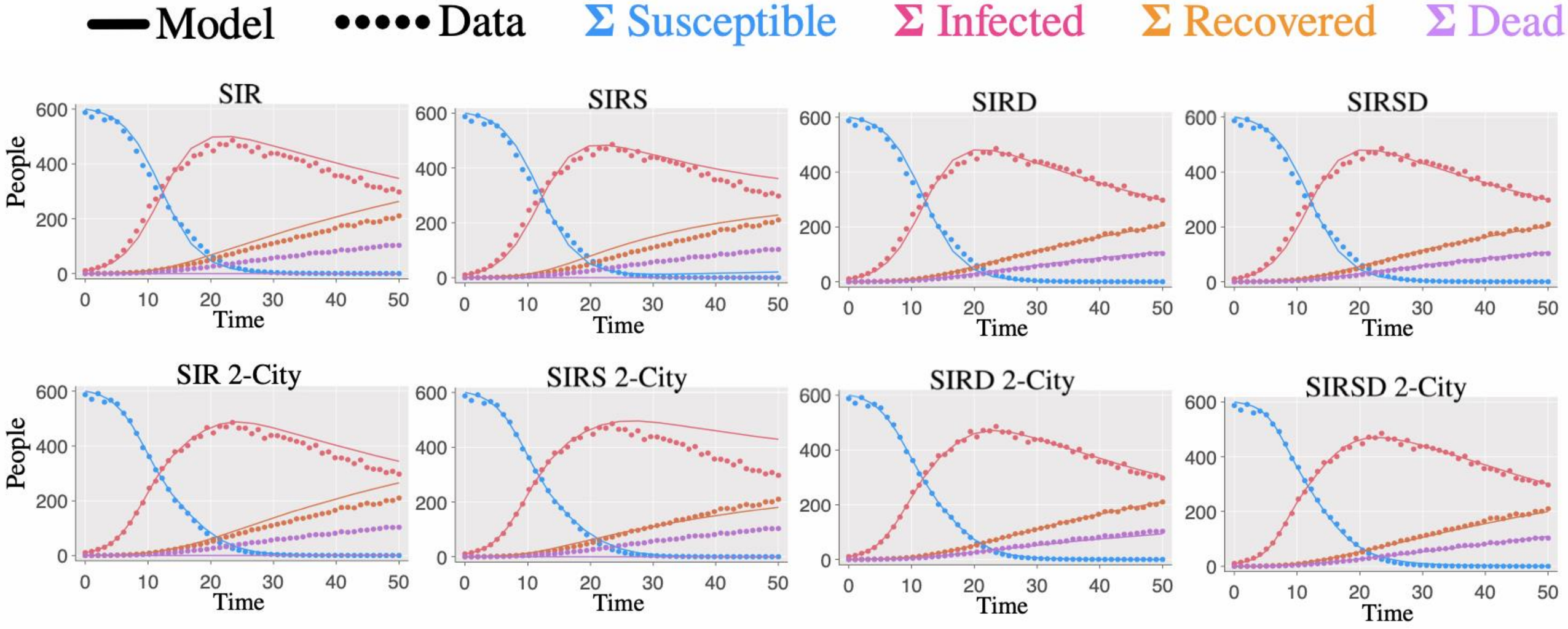
S: 90%, I: 10% and City X: 33%, City Y: 67%

For $(SIRD)_2$, we have S_X: 30%, D_X: 0%

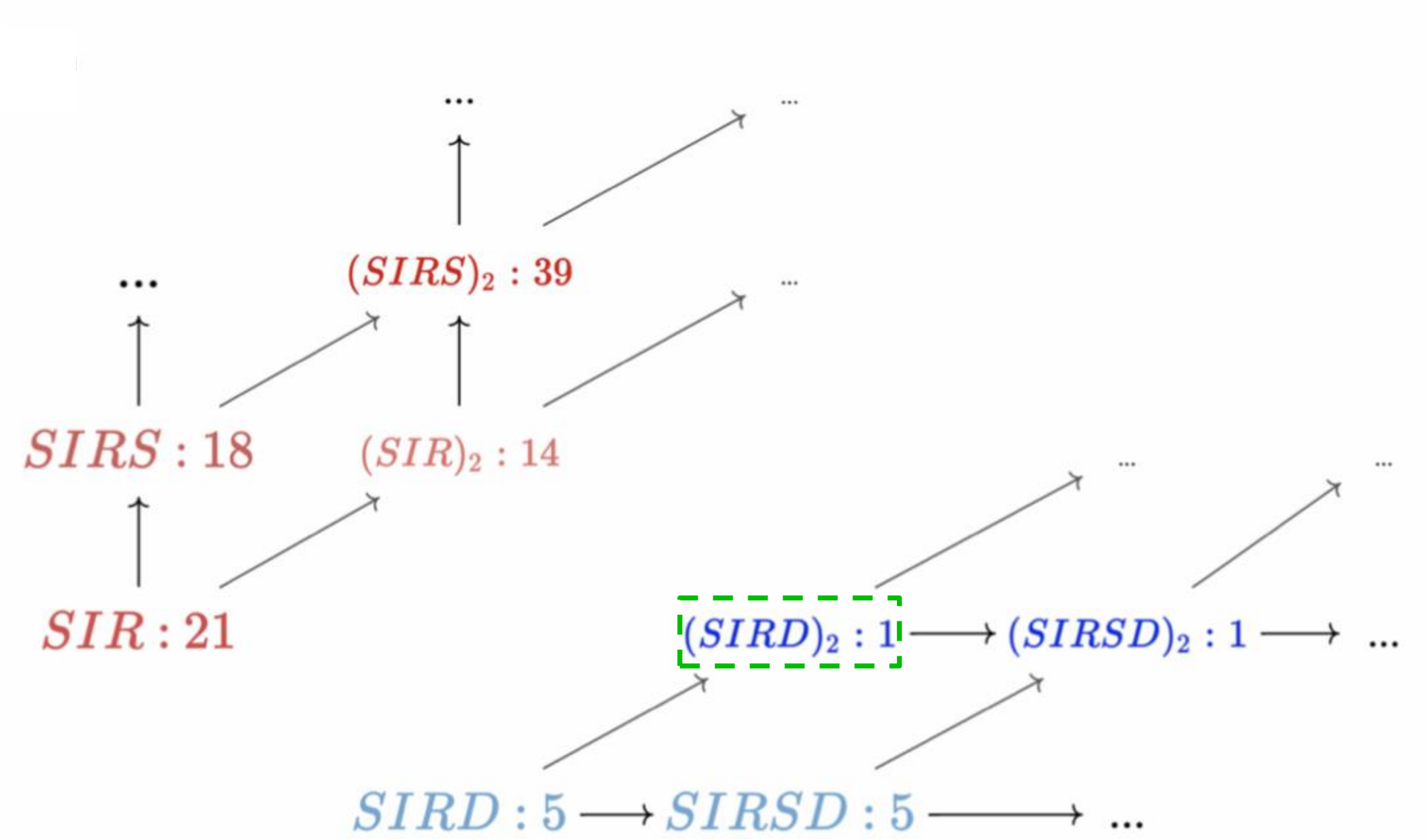
For $(SIRD)_2$, we have S_X: 10%, S_Y: 25%

S: 35%

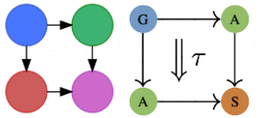
Model selection results



Model selection results



Future work



- More interesting “primitive” model space constructors
- Wiring diagram visualizer / GUI
- Lazy state space exploration
- Alternative applications as demonstrations (Boolean functions, circuits, NN)
- Hierarchical loss functions (optimizing overall goal + subgoals, together)

Thanks!

TOPOS INSTITUTE

T. Hanks

James Fairbanks

UF UNIVERSITY of FLORIDA

Sean Wu

IHME
Institute for Health Metrics and Evaluation

Andrew Baas

Georgia Tech Research Institute