Composing Stock and Flow Diagrams

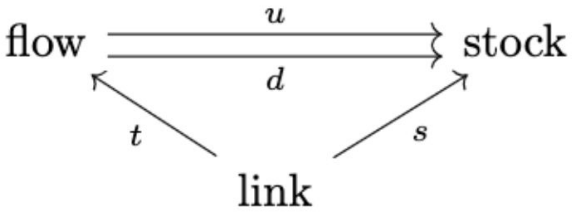
CANMOD Bootcamp 2022

Xiaoyan Li

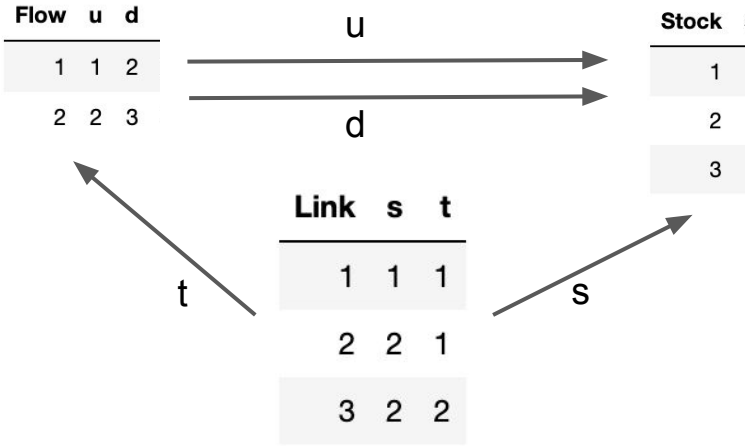
Outline

1. An example of a Stock and Flow diagram (Primitive schema)
2. Composing Stock and Flow Diagrams(Primitive schema)
    a. Structured Cospan
    b. uwd-algebra
3. Examples of composing Stock and Flow Diagrams(moderate complexity schema)

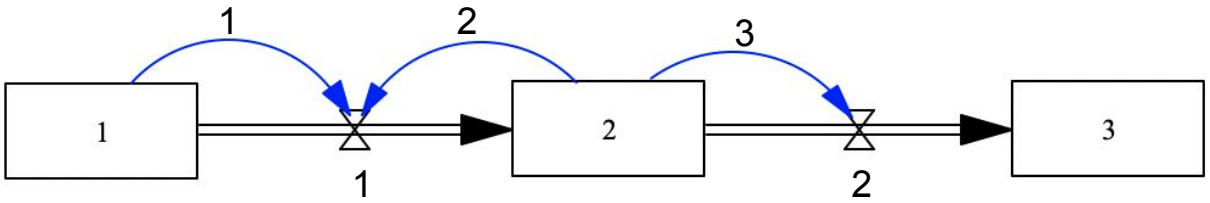
1.1 An example of a stock and flow diagram as CSet



Category C represents the primitive stock and flow diagram

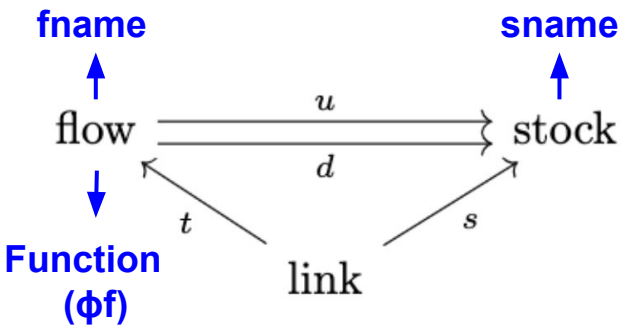


Functor from Category C to Set: C-Set



Primitive Stock and Flow Diagram

1.2 An example of a stock and flow diagram as ACSet



Category C, and Attributes

Flow	u	d	fname	ϕf
1	1	2	NewIncidence	fNewIncidence
2	2	3	NewRecovery	fNewRecovery

u

d

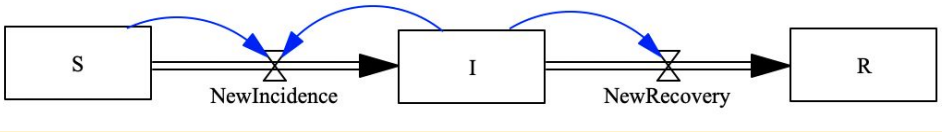
Stock	sname
1	S
2	I
3	R

t

s

Link	s	t
1	1	1
2	2	1
3	2	2

ACSet




```
# define functions ϕ of flows in the SIR model
fNewIncidence(u,p,t)=p.cβ*u.S*u.I/p.N
fNewRecovery(u,p,t)=u.I/p.tr
```

Stock and Flow Diagram, together with functions of flows

1.3 Codes of building a SIR model in StockFlow.jl

1. **SIR model**

```
# define functions ϕ of flows in the SIR model
fNewIncidence(u,p,t)=p.cβ*u.S*u.I/p.N
fNewRecovery(u,p,t)=u.I/p.tr

# StockAndFlowp(stocks,
#               (flow=>function, upstream=>downstream) => stocks linked)
sir = StockAndFlowp((:S, :I, :R),
   ((:NewIncidence=>fNewIncidence, :S=>:I)=>(:S,:I),
    (:NewRecovery=>fNewRecovery, :I=>:R)=>:I)
)
```

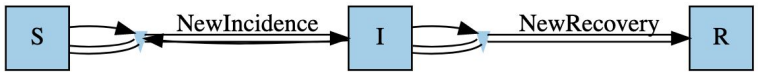
StockAndFlowp with elements Flow = 1:2, Stock = 1:3, Link = 1:3

Flow	u	d	fname	ϕf
1	1	2	NewIncidence	fNewIncidence
2	2	3	NewRecovery	fNewRecovery

Stock	sname
1	S
2	I
3	R

Link	s	t
1	1	1
2	2	1
3	2	2

```
Graph(sir)
```

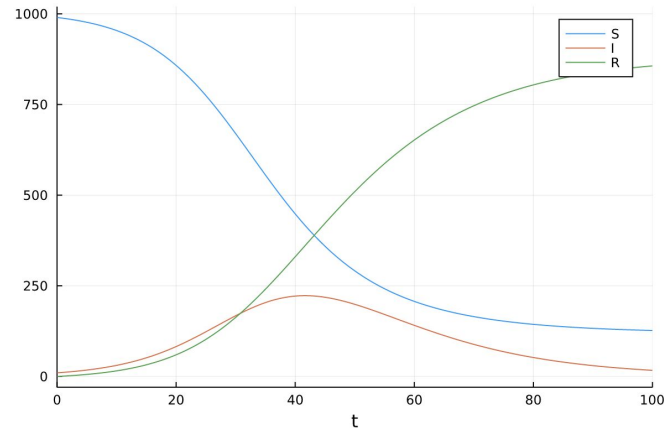


2.

```
# define constant parameters
p_sir = LVector(
    cβ=0.2, N=1000, tr=12
)
# define initial values for stocks
u0_sir = LVector(
    S=990, I=10, R=0
)
```

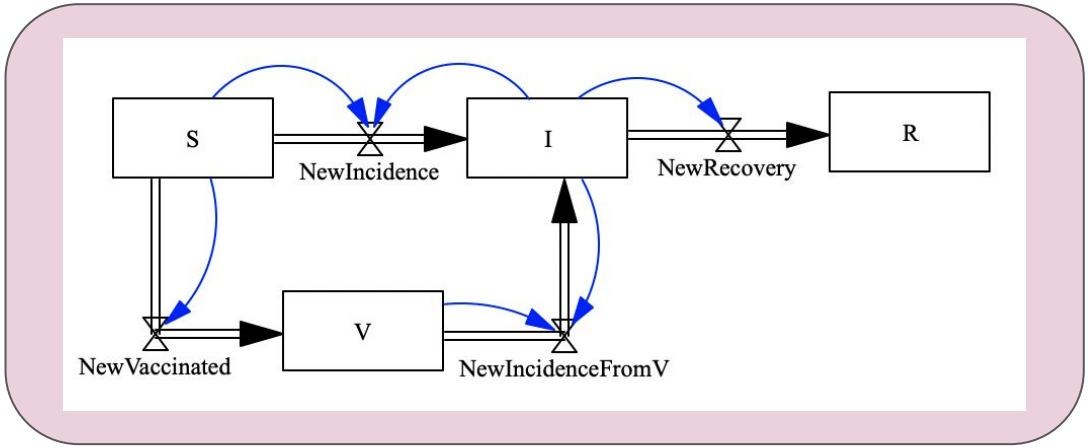
3.

```
# solve the ODEs
prob_sir = ODEProblem(vectorfield(sir),u0_sir,(0.0,100.0),p_sir);
sol_sir = solve(prob_sir,Tsit5(),abstol=1e-8);
plot(sol_sir)
```



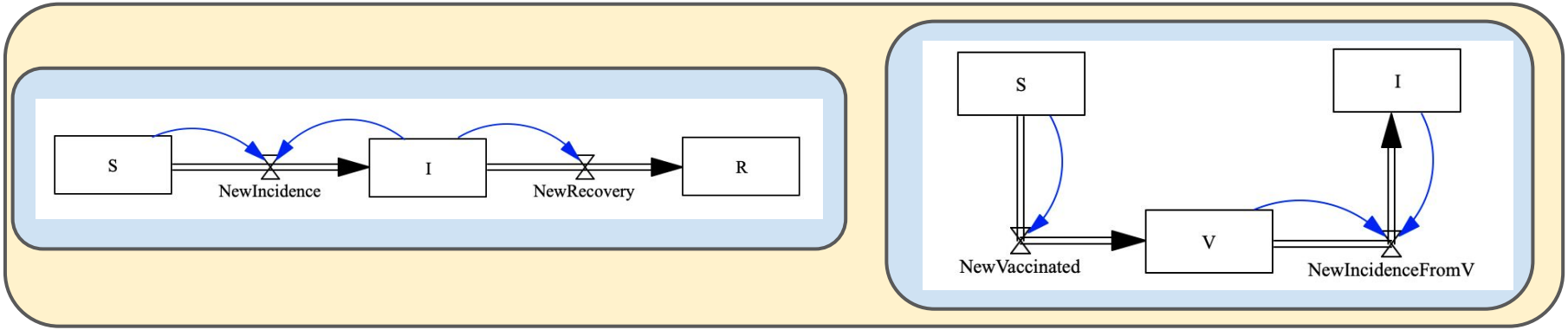
2. Composing Stock and Flow Diagrams

To Build:



Compose

We have:



2.1 Method 1: Decorated/Structured Cospan

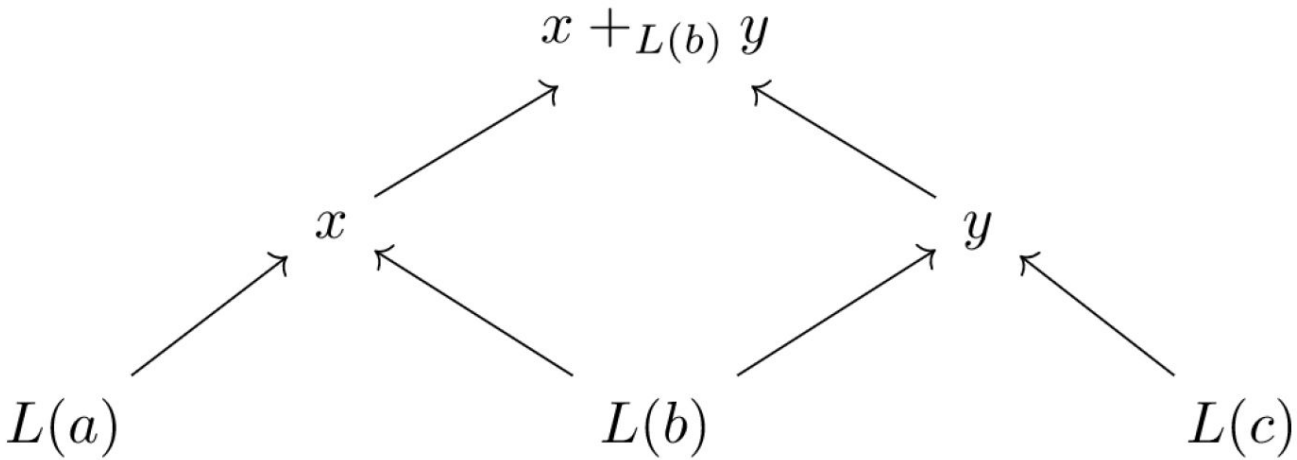
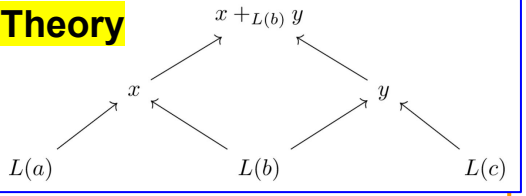


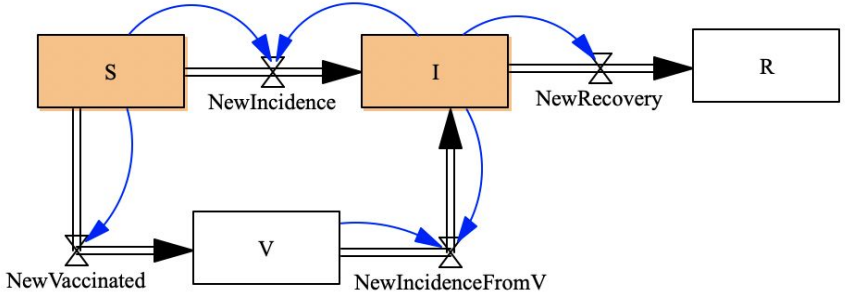
Figure cited from article in AlgebraicJulia blog, by *Micah Halter and Evan Patterson* https://www.algebraicjulia.org/blog/post/2020/10/structured-cospans/

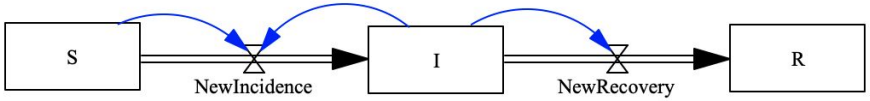
2.1 Method 1: Structured Cospan

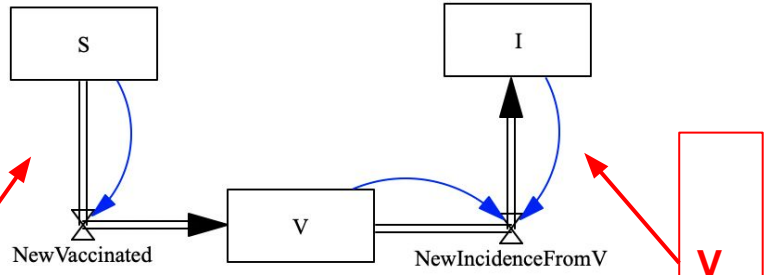
Theory



Example







R

a=1

L(a)= R

b=2

L(b)= S I

S

I

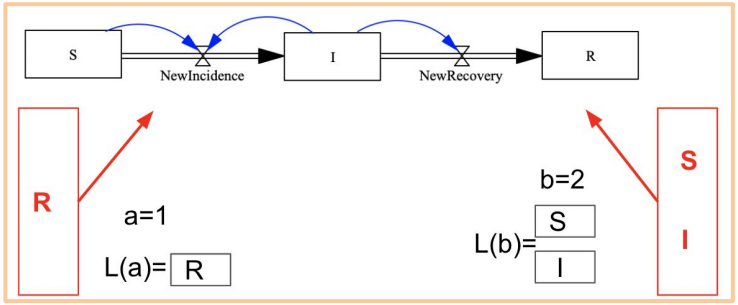
c=1

L(c)= V

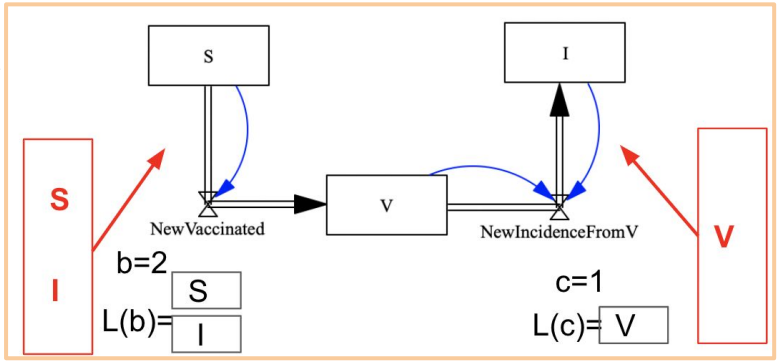
V

2.1 Method 1: Structured Cospan

```
open_sir=Open([:R],sir,[:S,:I]);
```




```
open_svi=Open([:S,:I],svi,[:V]);
```



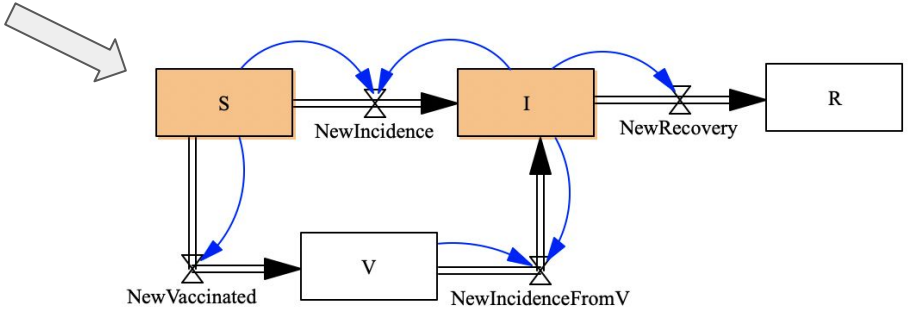

```
sirv1=apex(compose(open_sir,open_svi))
```

StockAndFlowp with elements Flow = 1:4, Stock = 1:4, Link = 1:6

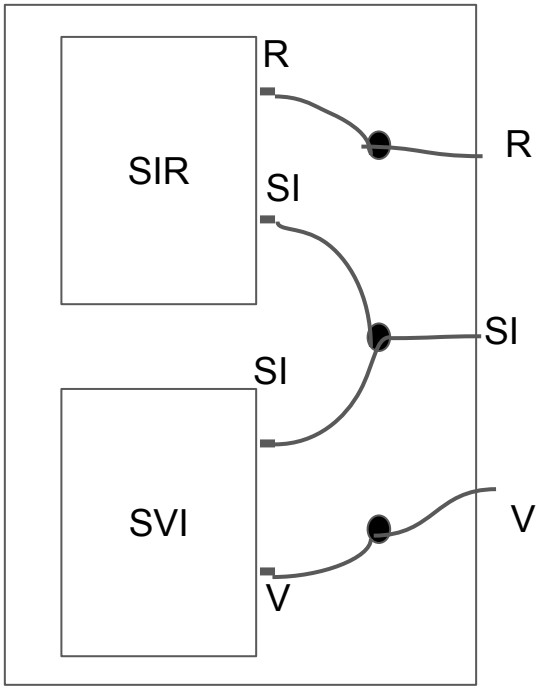
Flow	u	d	fname	ϕf
1	1	2	NewIncidence	fNewIncidence
2	2	3	NewRecovery	fNewRecovery
3	4	2	NewIncidenceFromV	fNewIncidenceFromV
4	1	4	NewVaccinated	fNewVaccinated

Link	s	t
1	1	1
2	2	1
3	2	2
4	4	3
5	2	3
6	1	4

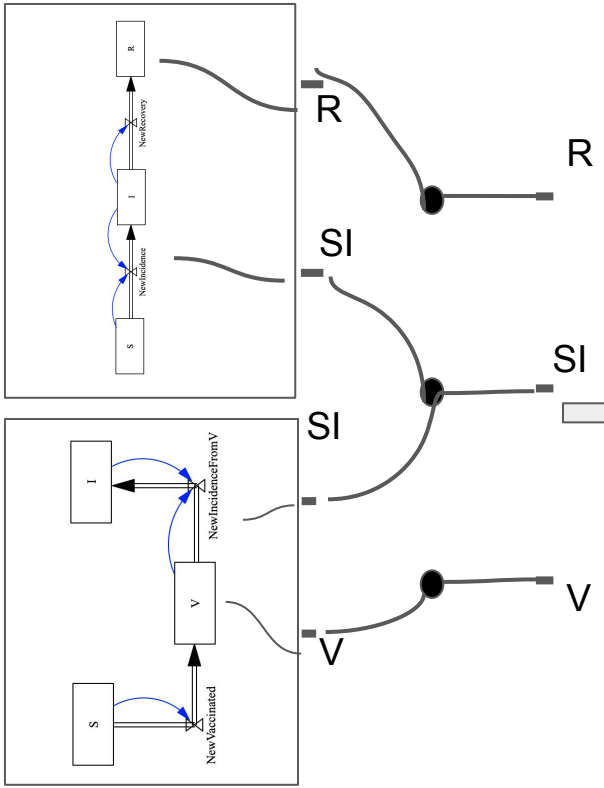
Stock	sname
1	S
2	I
3	R
4	V



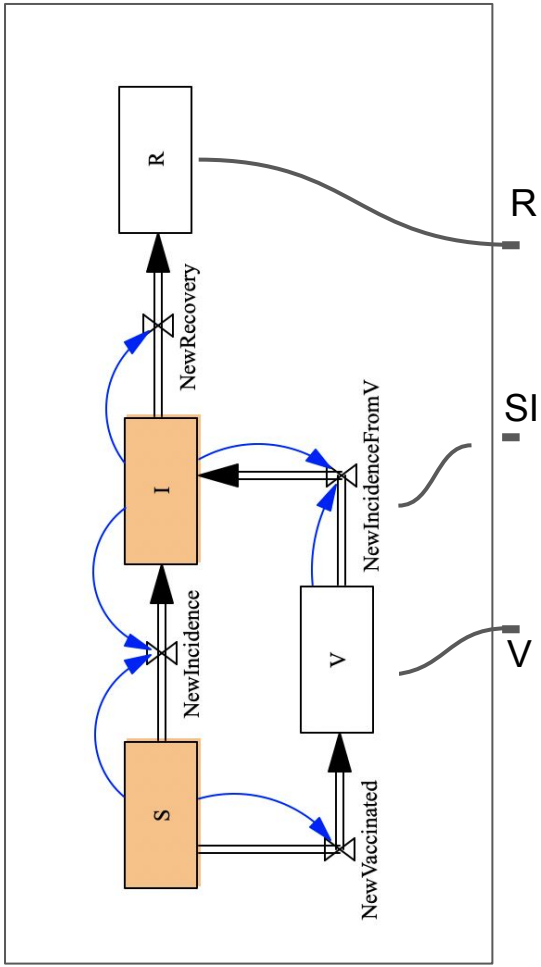
2.2 Method 2: UWD-algebra



Define composition rule: UWD



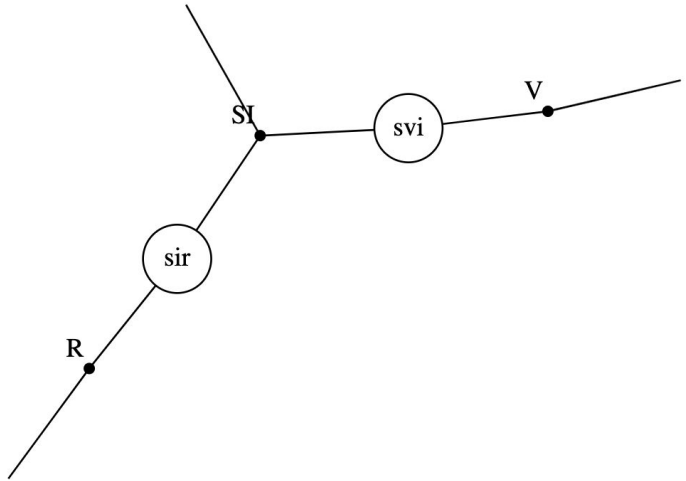
Fill in the open stock flow diagram



Composed model

2.2 Method 2: UWD-algebra codes

```
uwd_sirv = @relation (R, SI, V) begin
    sir(R,SI)
    svi(SI,V)
end;
display_uwd(uwd_sirv)
```



Define composition rule: UWD

```
sirv2=oapply(uwd_sirv,Dict(
        :sir=>Open(sir,[:R],[:S,:I]),
        :svi=>Open(svi,[:S,:I],[:V]))) |> apex
```

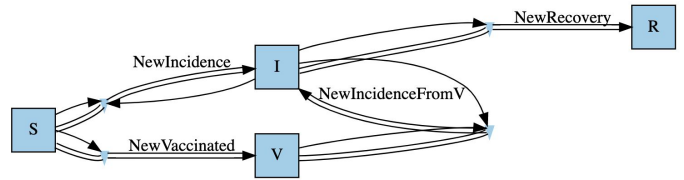
StockAndFlowp with elements Flow = 1:4, Stock = 1:4, Link = 1:6

Flow	u	d	fname	ϕf
1	1	2	NewIncidence	fNewIncidence
2	2	3	NewRecovery	fNewRecovery
3	4	2	NewIncidenceFromV	fNewIncidenceFromV
4	1	4	NewVaccinated	fNewVaccinated

Stock	sname
1	S
2	I
3	R
4	V

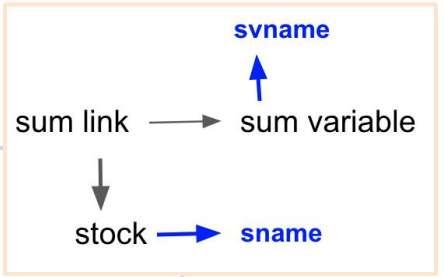
Link	s	t
1	1	1
2	2	1
3	2	2
4	4	3
5	2	3
6	1	4

```
Graph(sirv2)
```

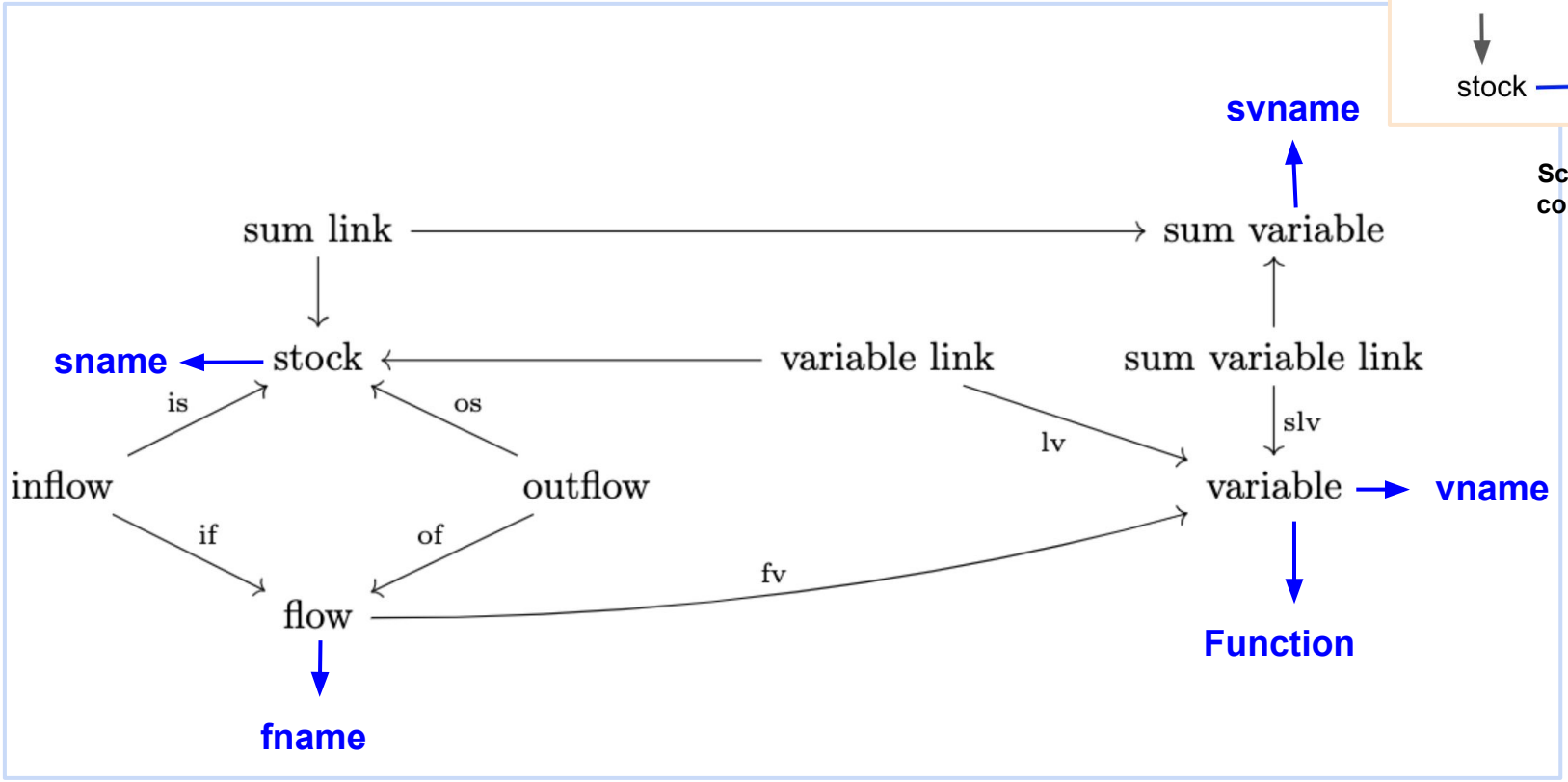


Apply the composition rule to the open stock and flow sub-components, and get the composed model

3. Examples in moderate complexity schema



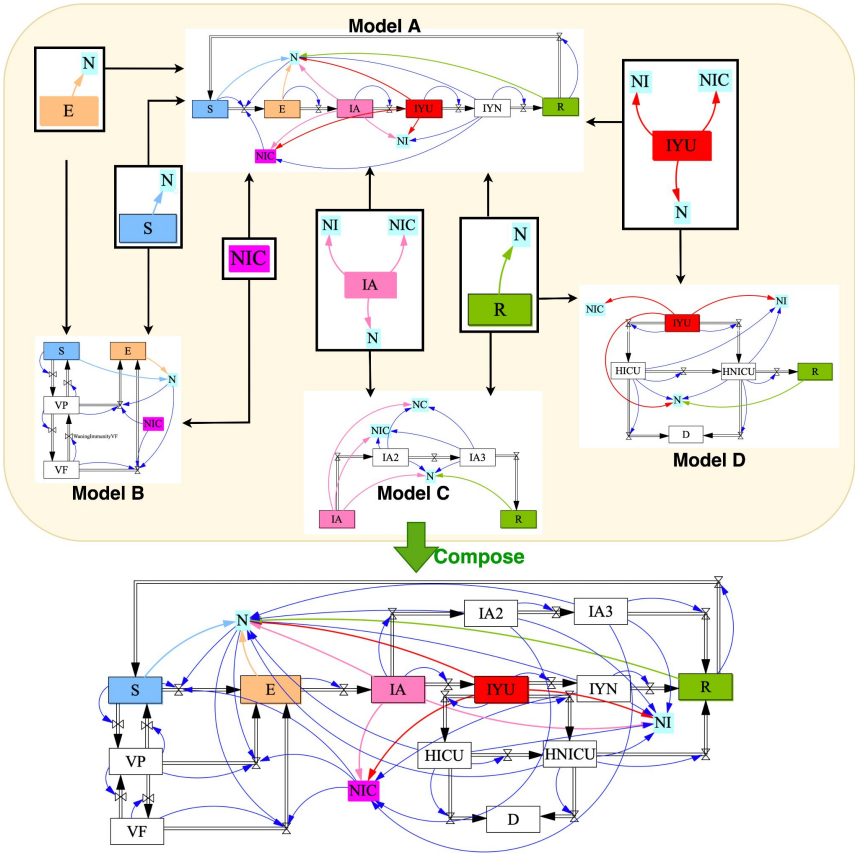
Schema of the common parts

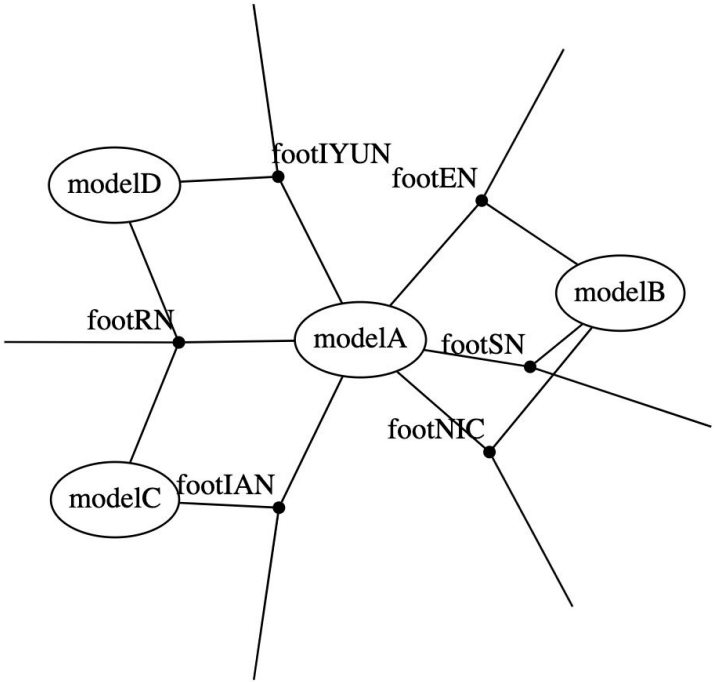


The Schema of Moderate Complexity Stock and Flow Diagram

3. Examples in moderate complexity schema

COVID-19 model, adapted from CEPHIL lab's work



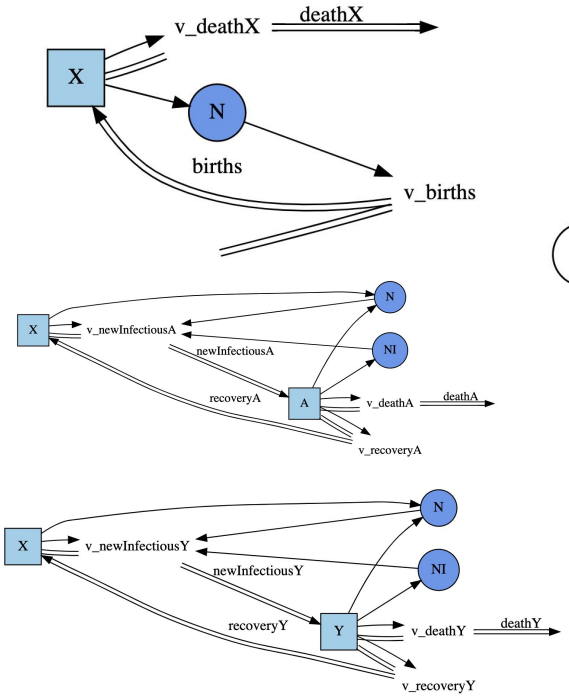


Composition rule defined in Catlab.jl

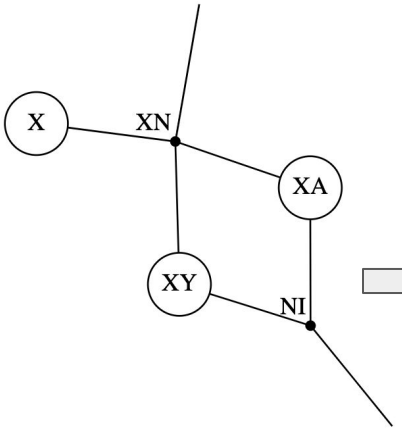
3. Examples in moderate complexity schema

Curable transmission model, adapted from Garnett's paper:

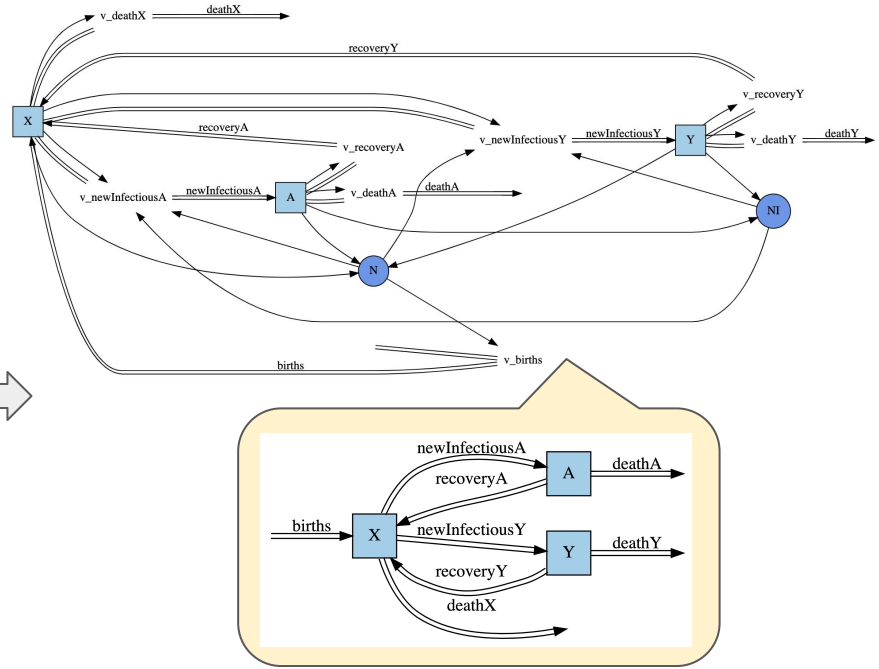
https://journals.lww.com/stdjournal/Fulltext/2000/11000/Epidemiology_and_Control_of_Curable_Sexually.7.aspx



3 sub-components



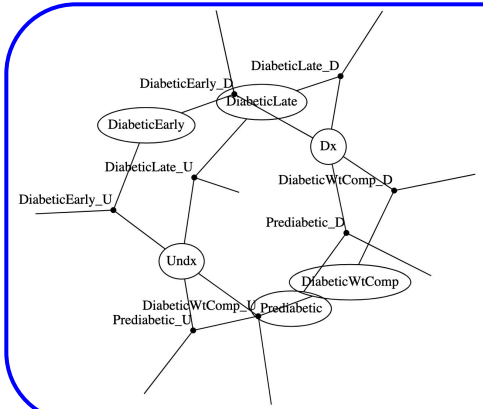
Composition rule

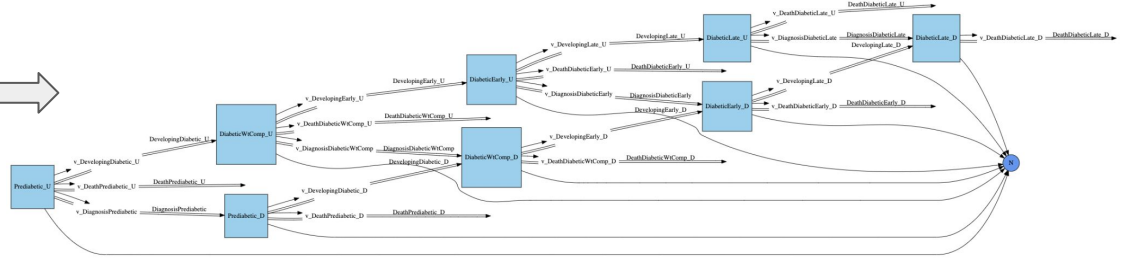


Composited model

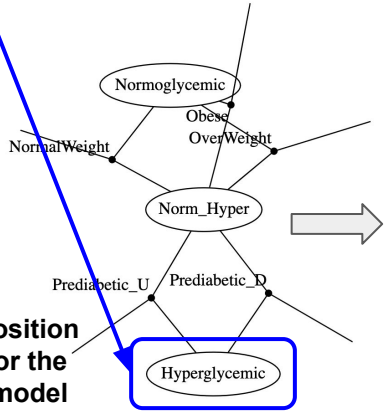
3. Examples in moderate complexity schema

Diabetes model, adapted from Zhang's thesis: https://harvest.usask.ca/handle/10388/ETD-2011-07-56

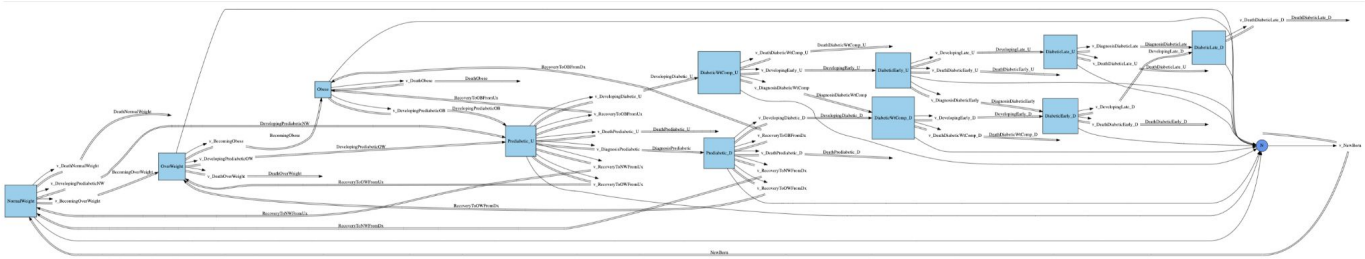




Composition rule for the Hyperglycemic Population model



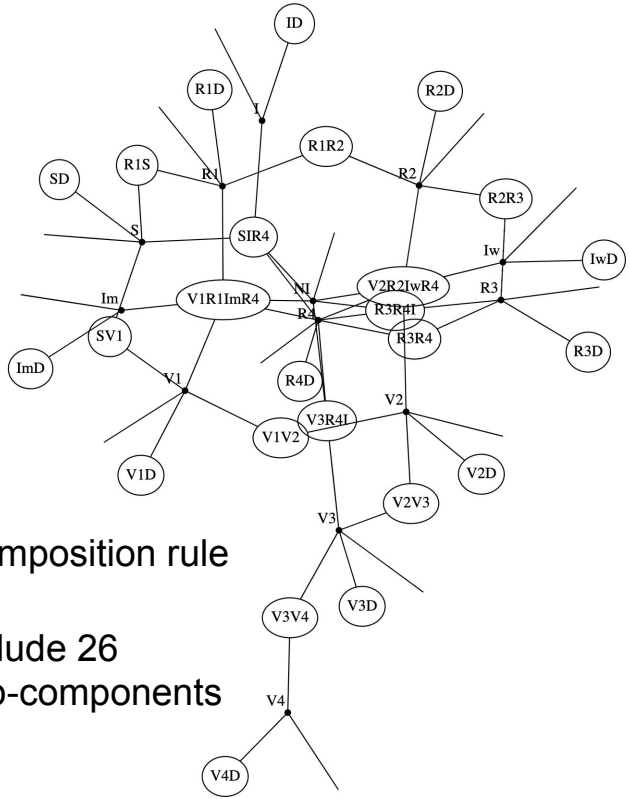
Composition rule for the final model



Final composed diabetes model

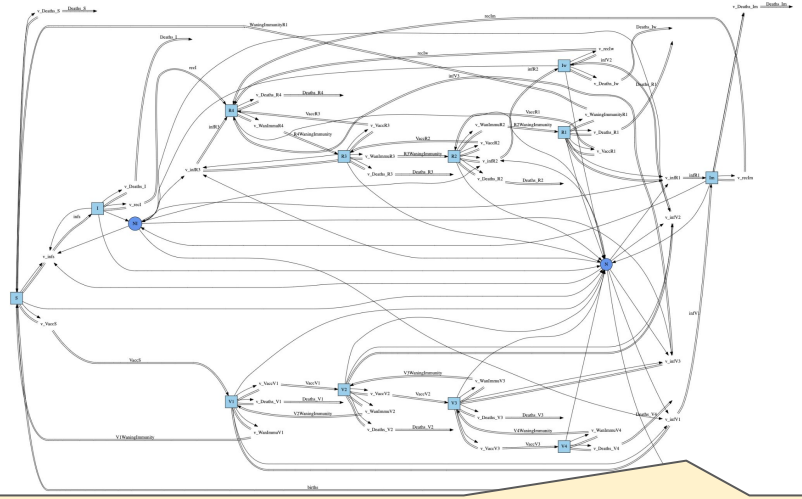
3. Examples in moderate complexity schema

Pertussis model, adapted from Hethcote's paper:
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.532.5615&rep=rep1&type=pdf

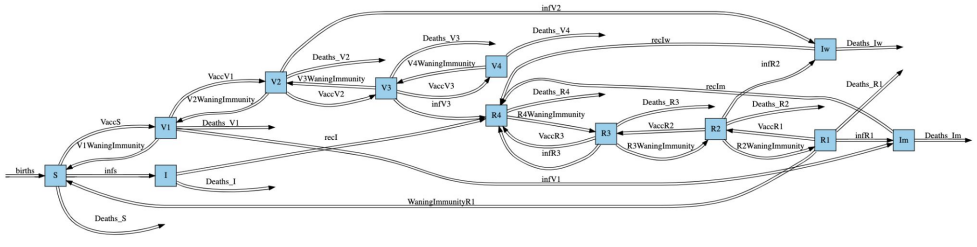


Composition rule

Include 26 sub-components



```
Graph(pertussis_model,type="SF")
```



Priorities for Future Work

- Introducing analysis-transparent functions
- Broadening permissible connections
- Unit & dimensional attributes
- Modular stratification via pullbacks
- Adding supports for additional semantics
  - Unit & dimensional inference & correctness, parameter space reduction via Buckingham π Thm
  - Calibration
  - Computational statistics methods (Particle filtering, Particle MCMC)
  - Stochastic simulation
  - Dynamic analysis mechanisms (Loop gains over time, eigenvalue elasticity over time, …)
- Collaborative graphical interface
  - Synchronization with evolving capabilities of Algebraic Julia support (including composition, alternative semantics, stratification, etc.)
  - Version control
  - QA support: Issue tracking, testability & mocking, etc.