Open Systems

What?

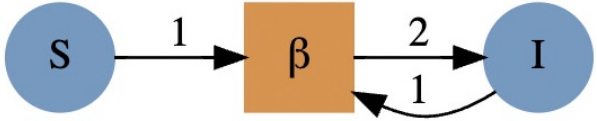
Systems that can interact with their environment

Why?

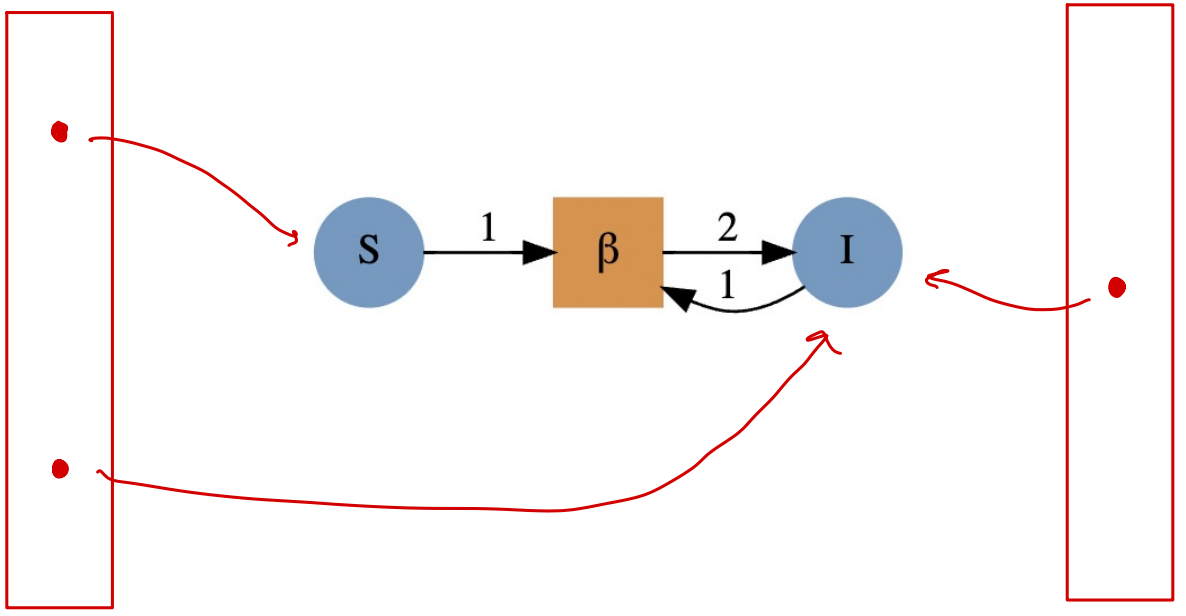
- Real-world systems interact
- Divide and conquer

Open Systems

Petri nets

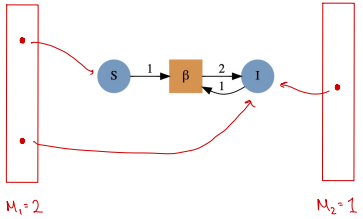


Open Petri nets

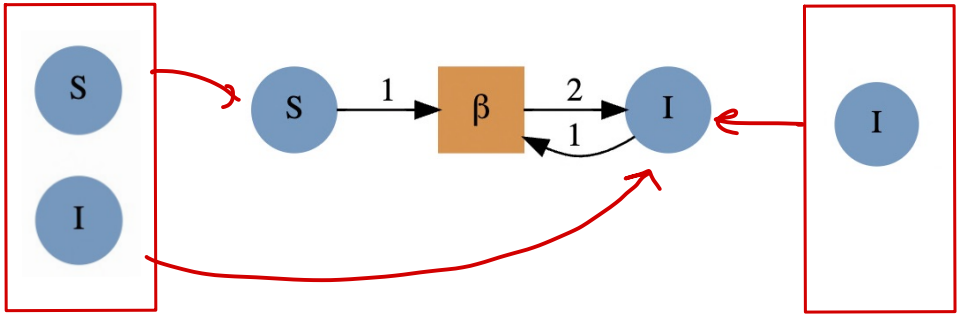


Open Systems

Another perspective on



A morphism between C-sets



A Petri net with no transitions

$M_1 = 2$

$M_2 = 1$

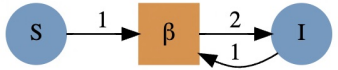
Open Petri Nets

```
using Catlab, Catlab.CategoricalAlgebra
using AlgebraicPetri
```

Infection Process

First we define a (closed) Petri net representing an infection process.

```
inf_net = LabelledPetriNet([:S, :I],
    :β => ((:S, :I) => (:I, :I))
)

Graph(inf_net)
```

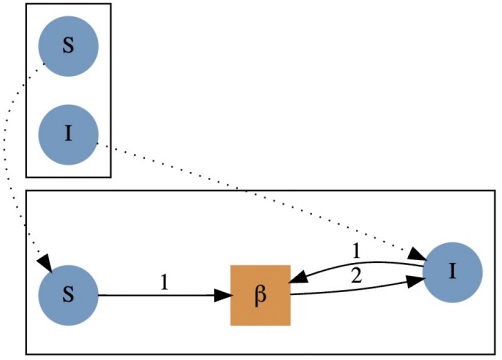


We turn this Petri net into an *open* Peti net by specifying two legs and the species that they expose.

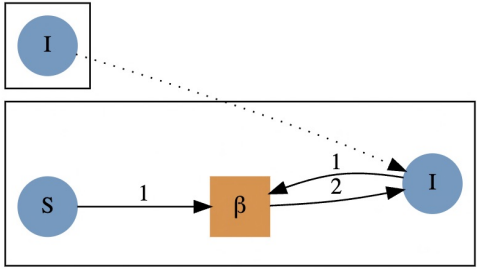
```
open_inf = Open([:S, :I], inf_net, [:I]);
```

We can visualize the two legs as follows.

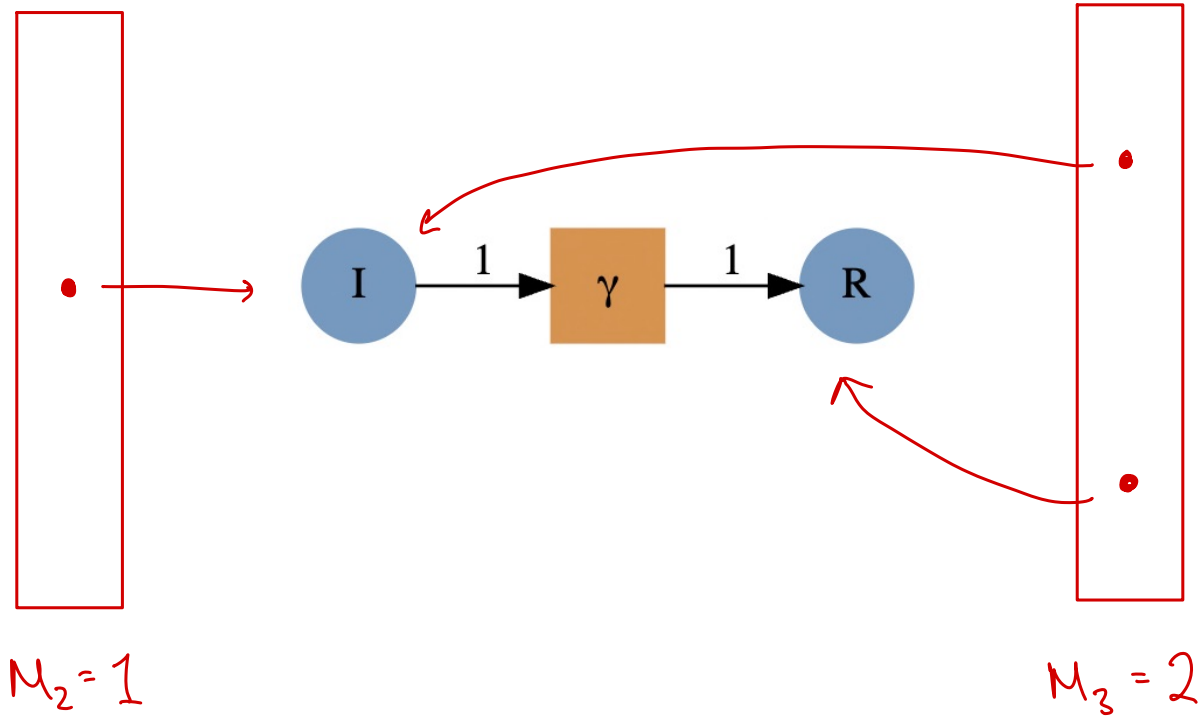
```
Graph(legs(open_inf)[1])
```




```
Graph(legs(open_inf)[2])
```



Open Petri nets

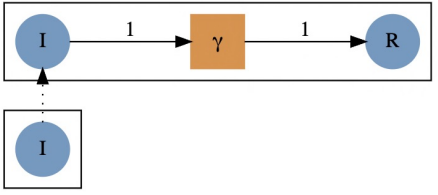


Recovery Process

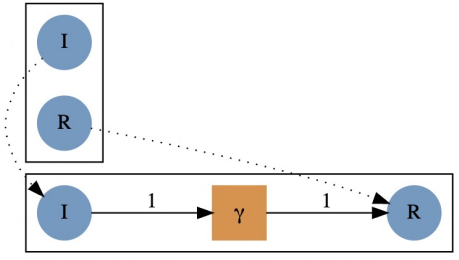
Let's repeat these steps for a recovery process.

```
rec_net = LabelledPetriNet([:I, :R],
    :γ => (:I => :R)
)

open_rec = Open([:I], rec_net, [:I, :R])

Graph(rec_net)
```


```
Graph(legs(open_rec)[1])
```



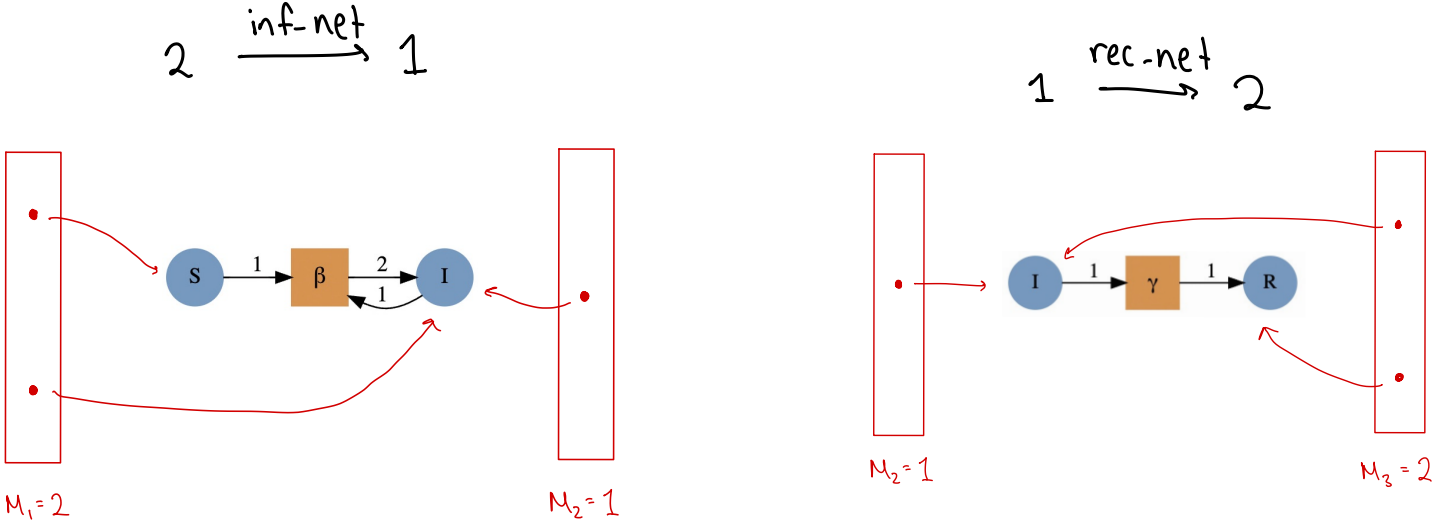

```
Graph(legs(open_rec)[2])
```



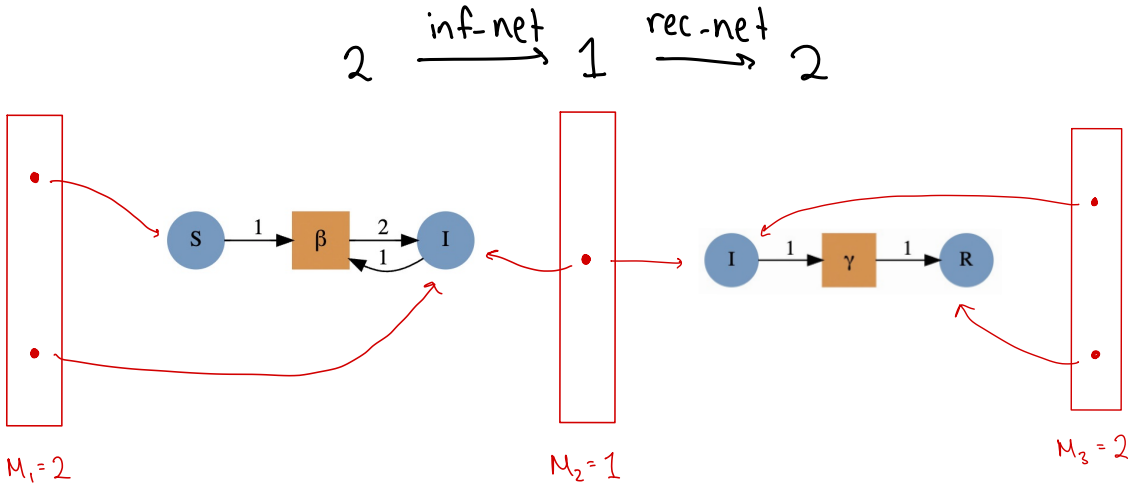
Open Systems

We have a category called **OpenPetri** with

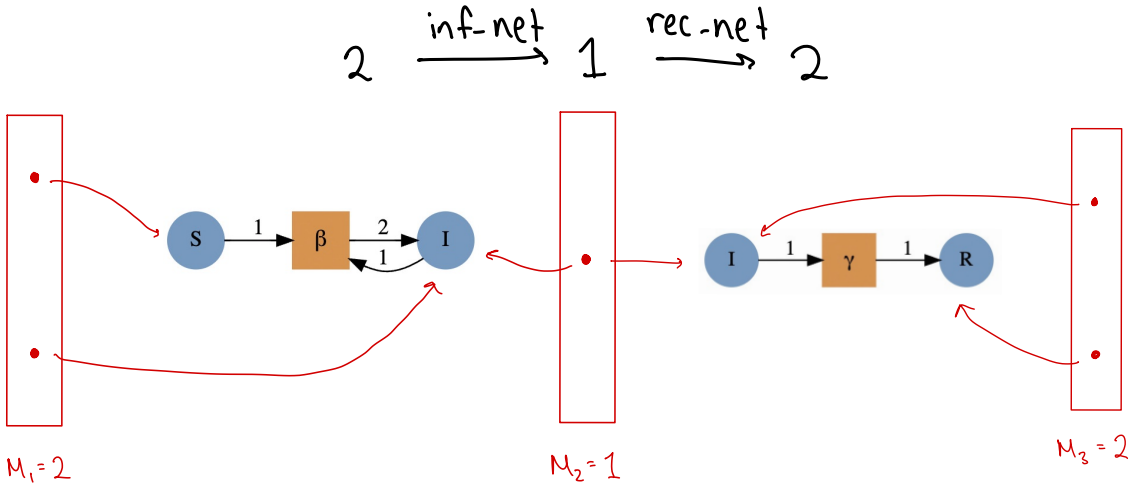
- objects : finite sets
- morphisms : open Petri nets



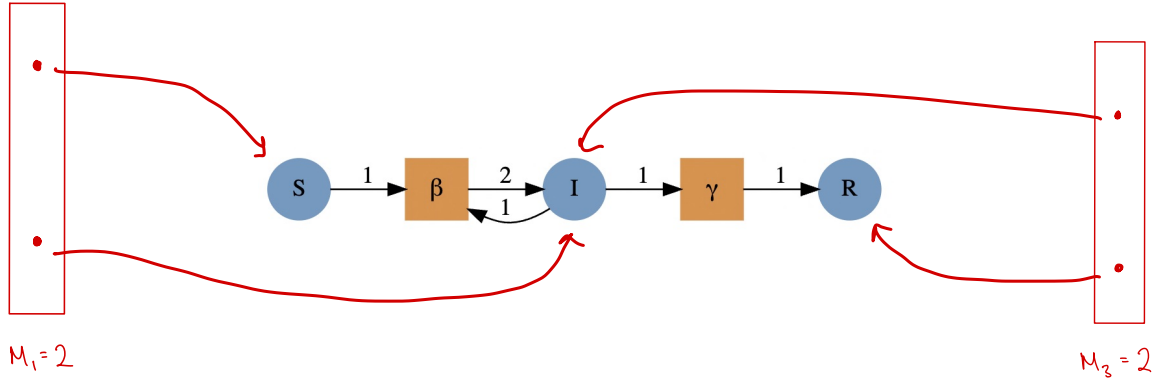
Composing Open Systems



Composing Open Systems



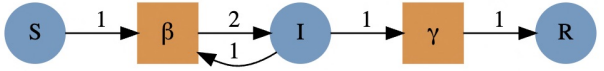
=



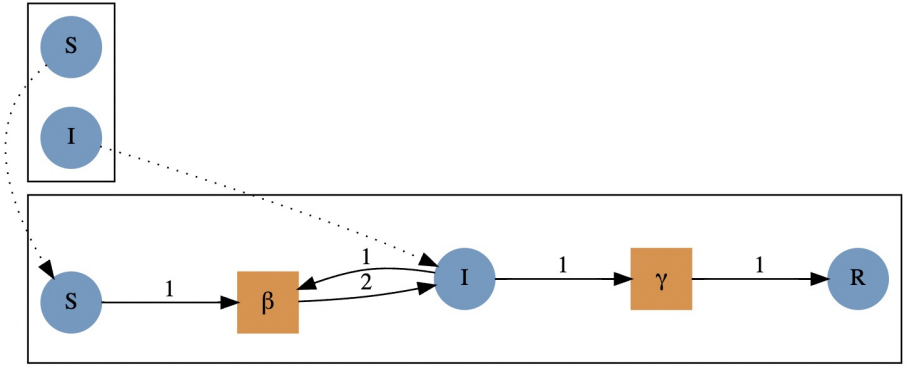
Composing Open Systems

Now we want to compose our two open systems `open_inf_net` and `open_rec_net`.

```
open_sir = compose(open_inf, open_rec);
Graph(open_sir)
```

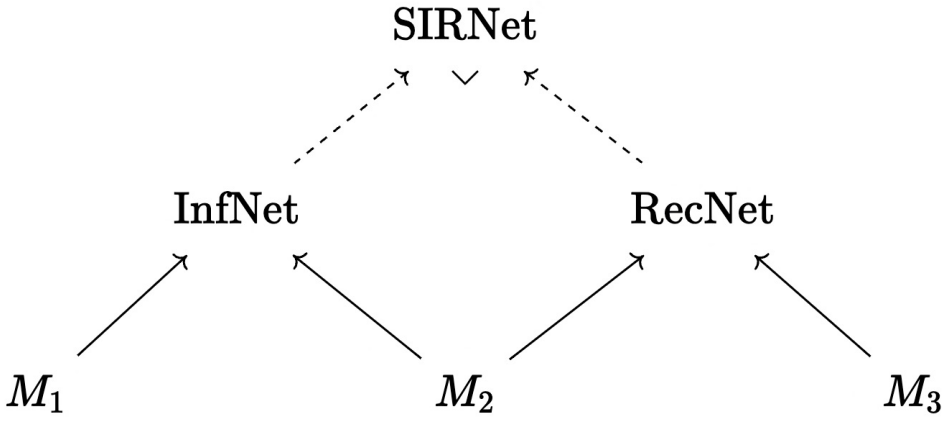



```
Graph(legs(open_sir)[1])
```



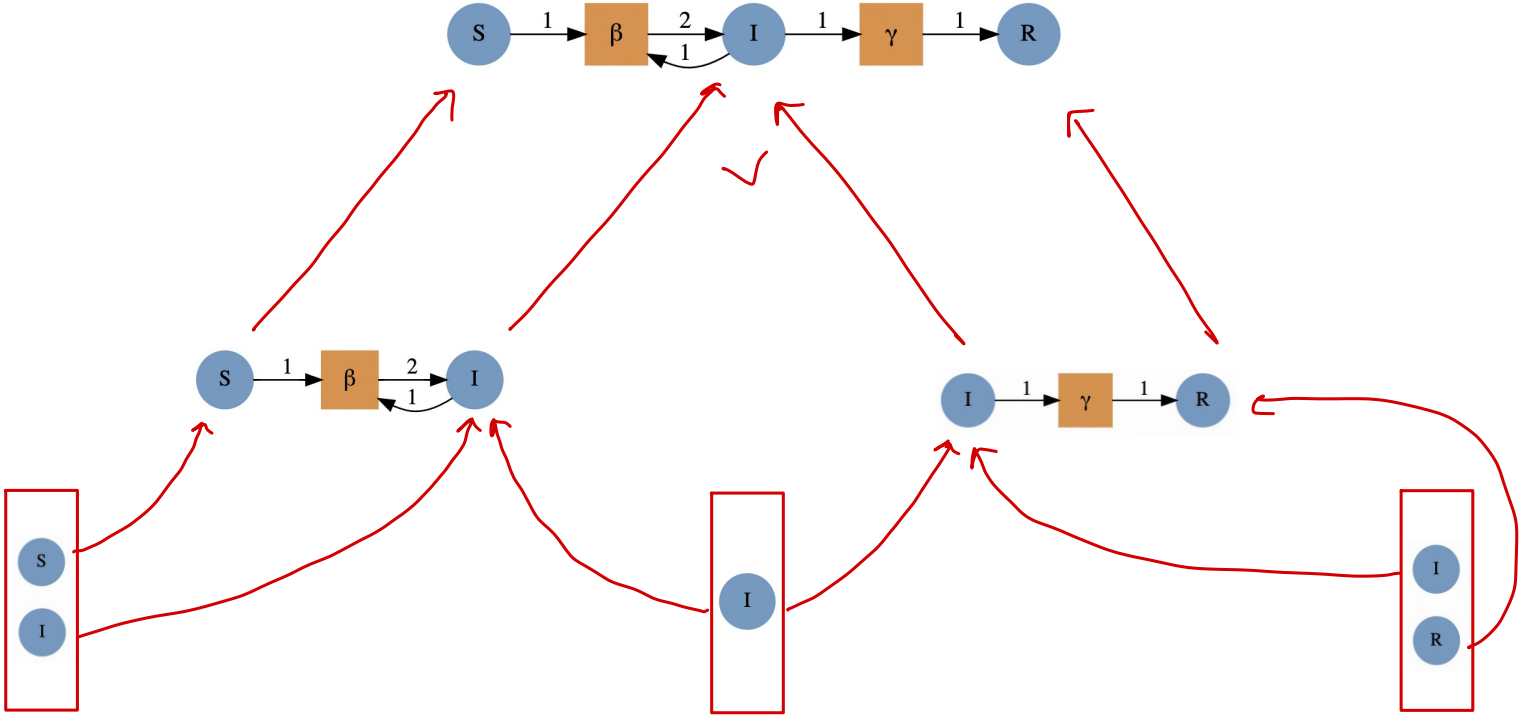
Composing Open Systems

Composing structured cospans with pushouts



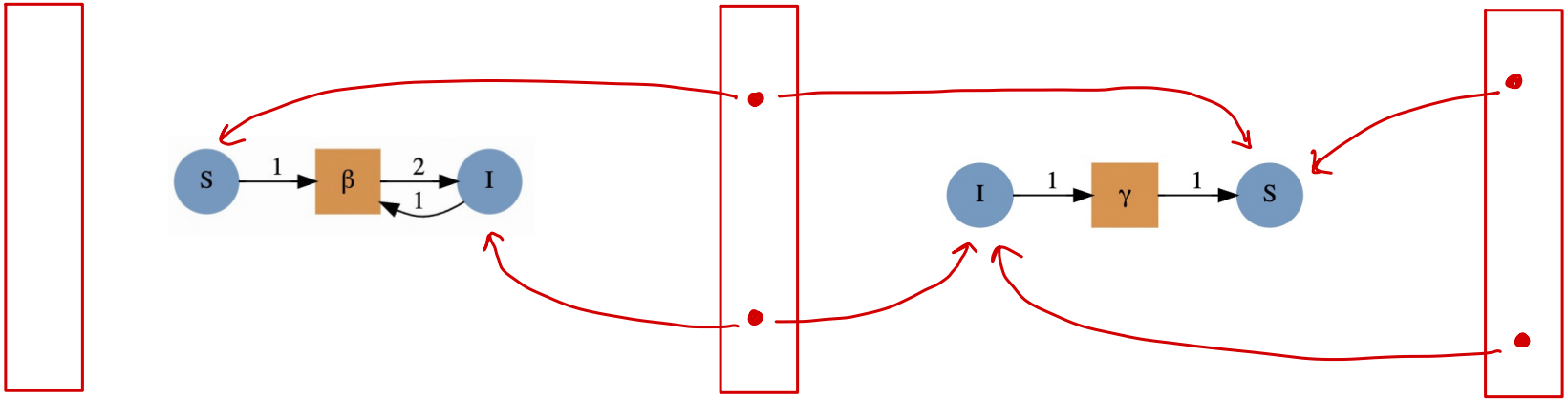
Composing Open Systems

Composing structured cospans with pushouts



Composing Open Systems

your turn! Implement this composition



0 ⟶ 2 ⟶ 2