Structural Operational Semantics for Graph Rewriting[1]

Andrei Dorman
Roma Tre, LIPN

Tobias Heindel
LIPN

ICE 2011

June 9th, 2011

[1] This work was partially supported by grants from Agence Nationale de la Recherche, ref. ANR-08-BLANC-0211-01 (COMPLICE project) and ref. ANR-09-BLAN-0169 (PANDA project).

Purpose Inductive definition of Labeled Transition Systems (LTS) for graph transformation (GTS).

Why? To see how similar graph rewriting is to process calculi.

Problem Not at all trivial in GTS as it is usually in process calculi.

But possible under some conditions.

Graph transformation and LTS

- General computation models:
  - States modeled by suitable graphs
  - State changes by graph transformations
    → behavior can be expressed by a transition system.

- Open systems vs. Closed systems
  - Environment
  - Internal action vs. external action

- Labelled Transition Systems:
  Automatic derivation techniques

Rewriting rule - example

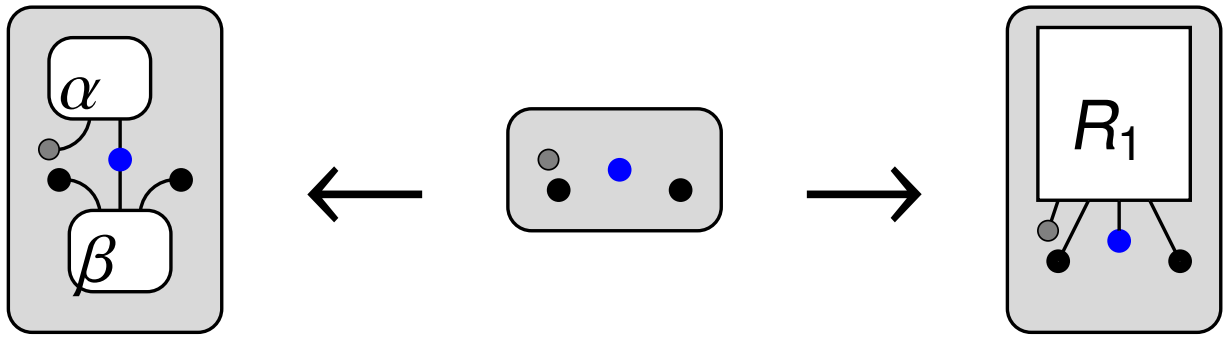
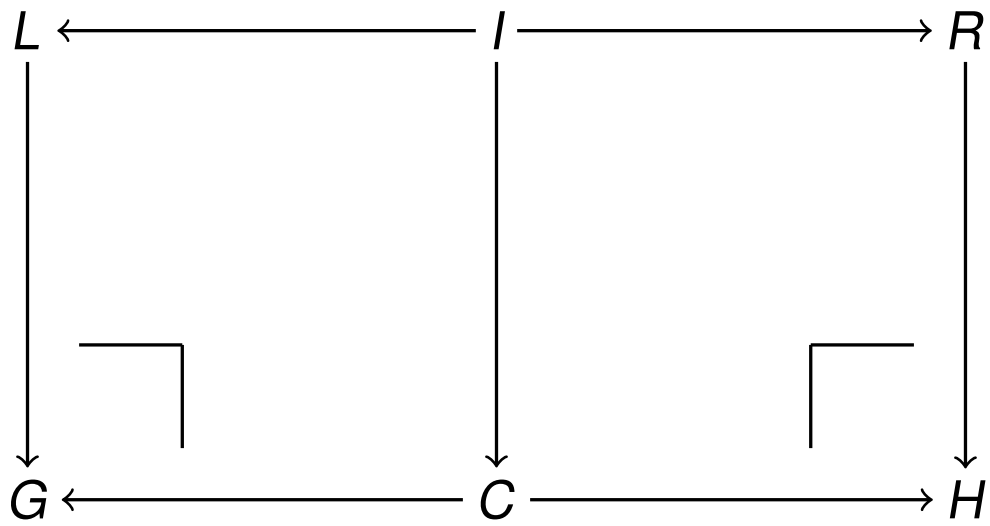


Figure: Reaction rules of a hypergraph transformation system $\mathcal{S}$.

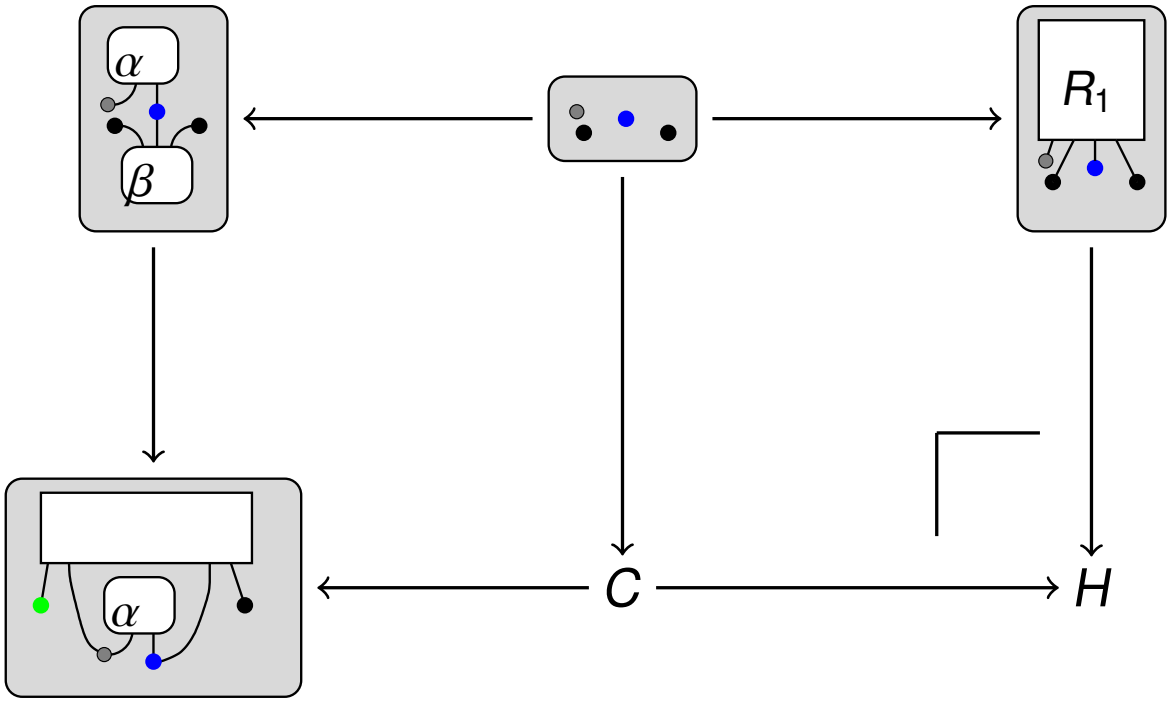
Double-Pushout

How does one rewrite?



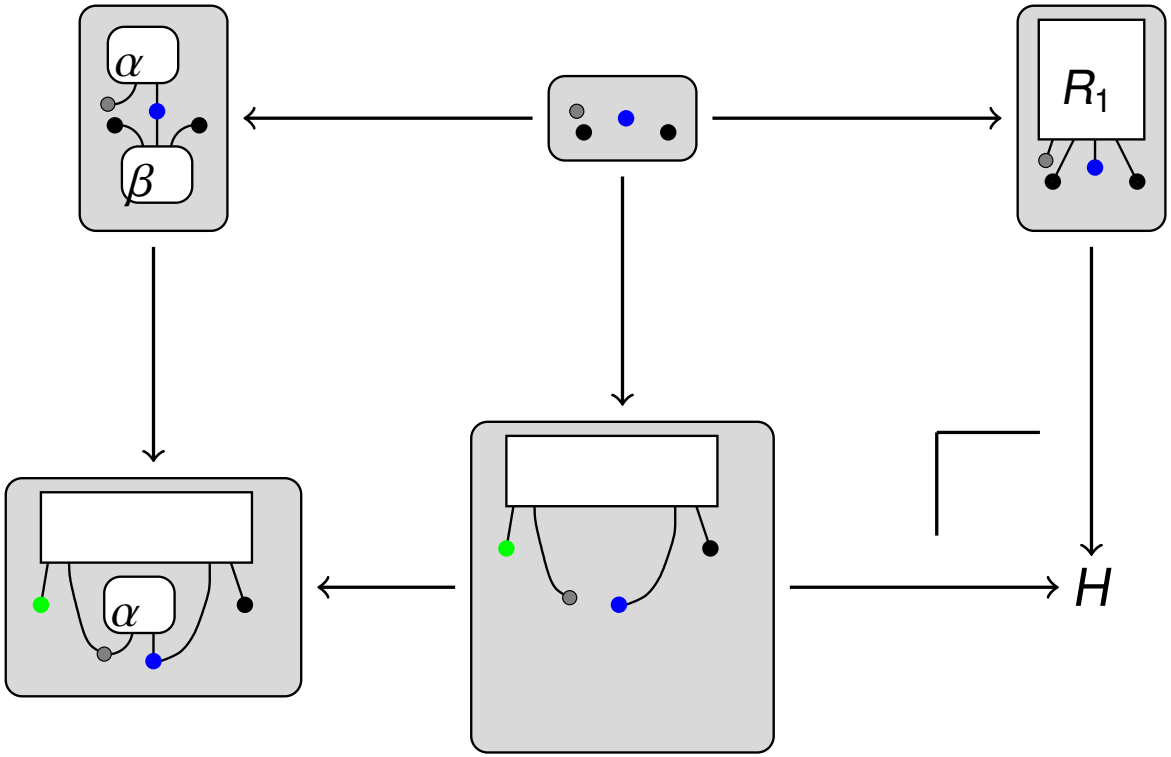
Double-Pushout

How does one rewrite?



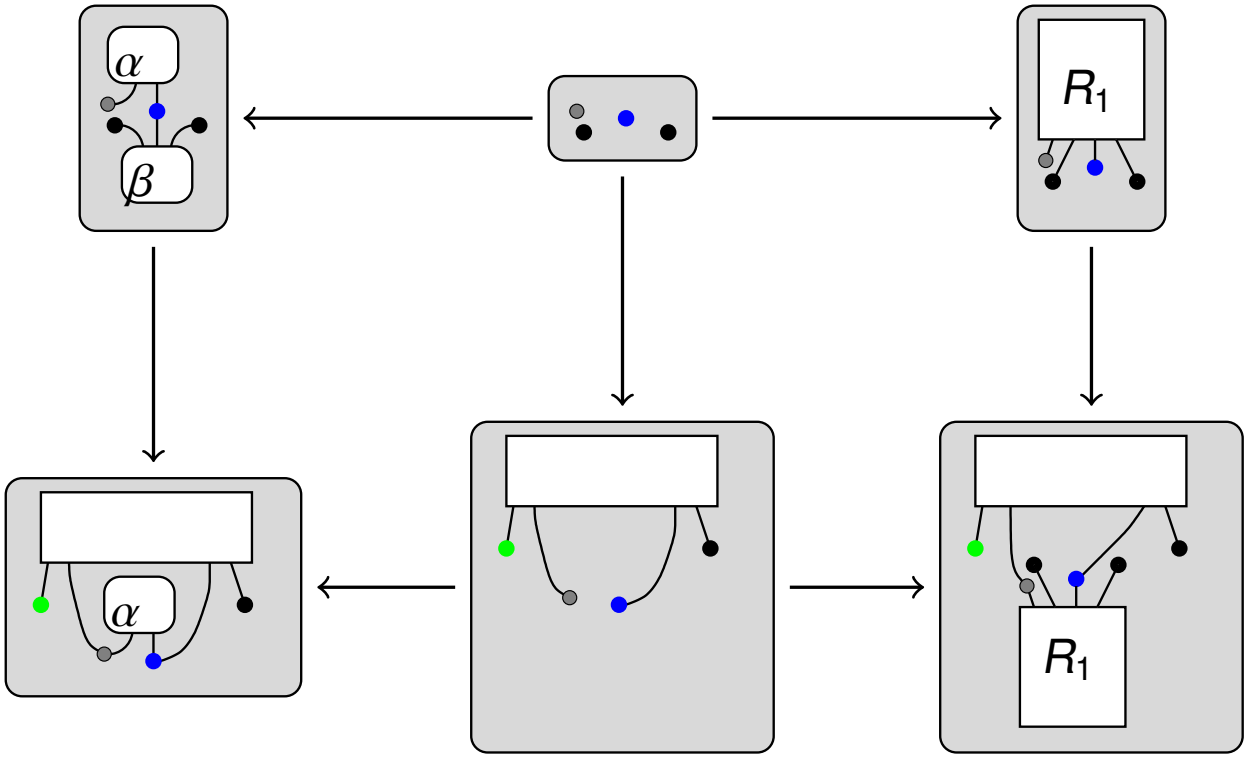
Double-Pushout

How does one rewrite?



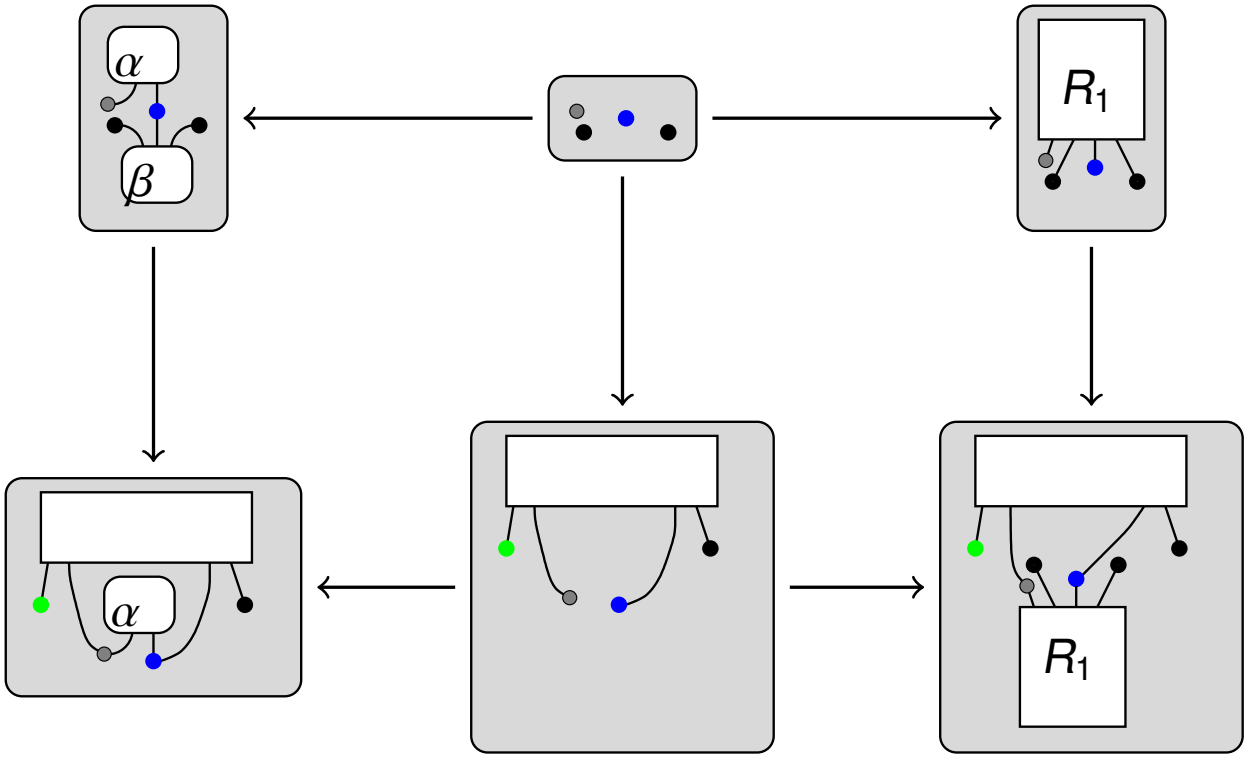
Double-Pushout

How does one rewrite?



Double-Pushout

How does one rewrite?



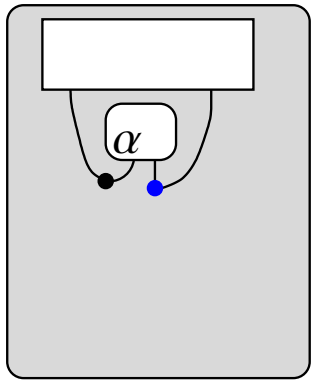
That's internal action.

External actions?

Waiting “something” from outside: we are missing something

≡ we don’t own the whole L graph.

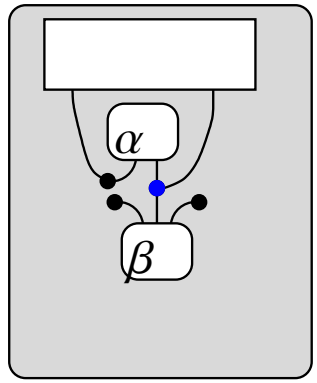
Idea Borrow from the context whatever is missing.



External actions?

Waiting “something” from outside: we are missing something
≡ we don’t own the whole L graph.

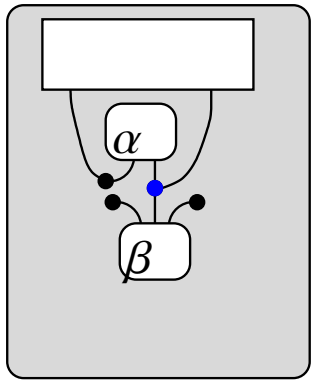
Idea Borrow from the context whatever is missing.



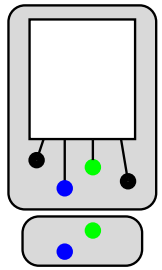
External actions?

Waiting “something” from outside: we are missing something
≡ we don’t own the whole L graph.

Idea Borrow from the context whatever is missing.



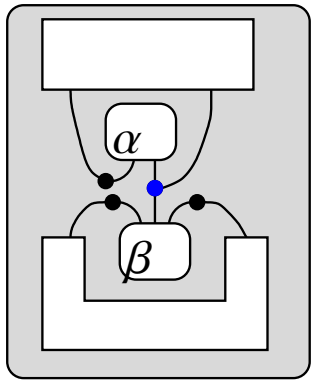
- Interface: states are injections of graphs
- Too many contexts...: minimality



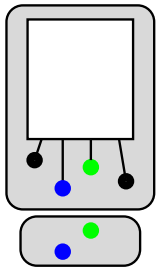
External actions?

Waiting “something” from outside: we are missing something
≡ we don’t own the whole L graph.

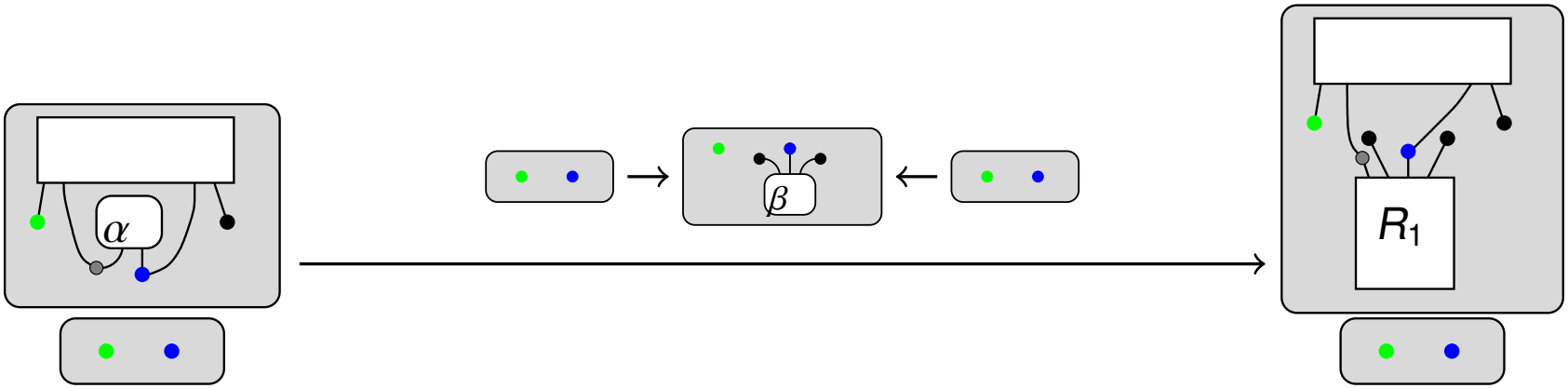
Idea Borrow from the context whatever is missing.



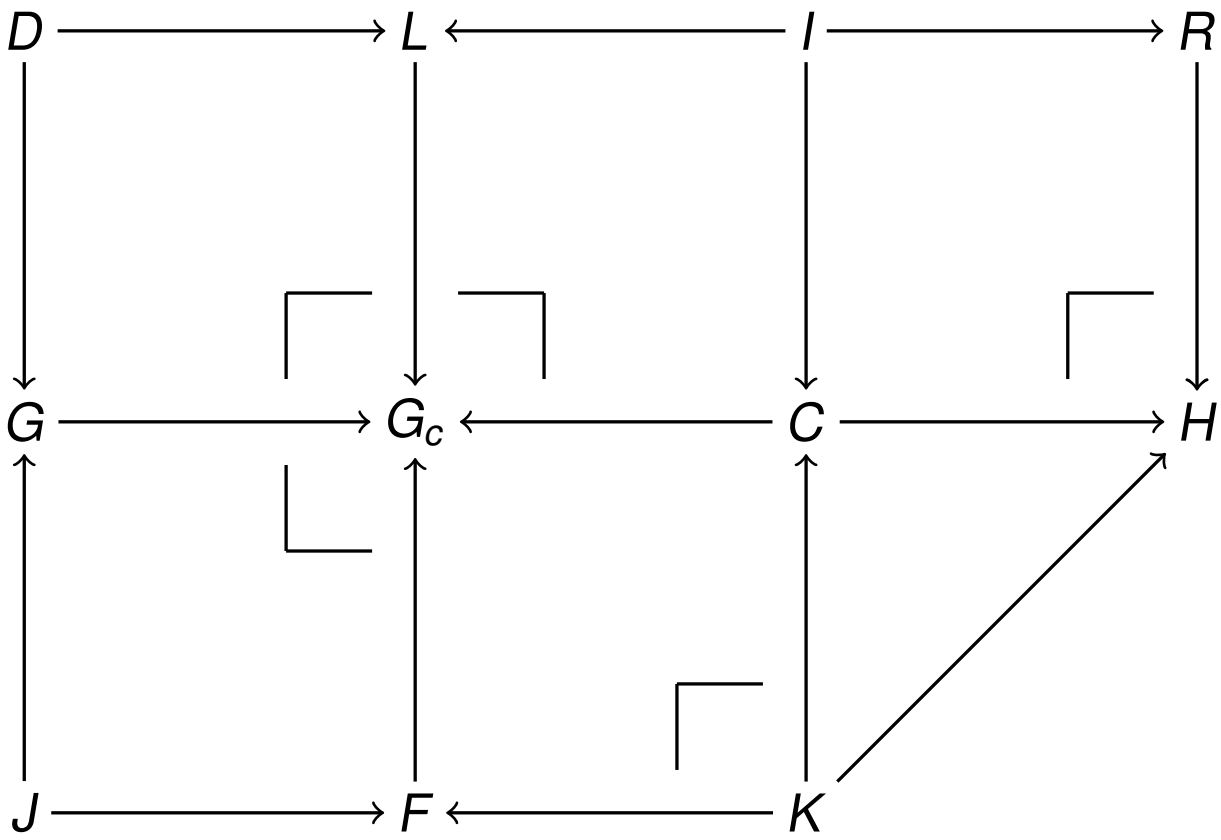
- Interface: states are injections of graphs
- Too many contexts...: minimality



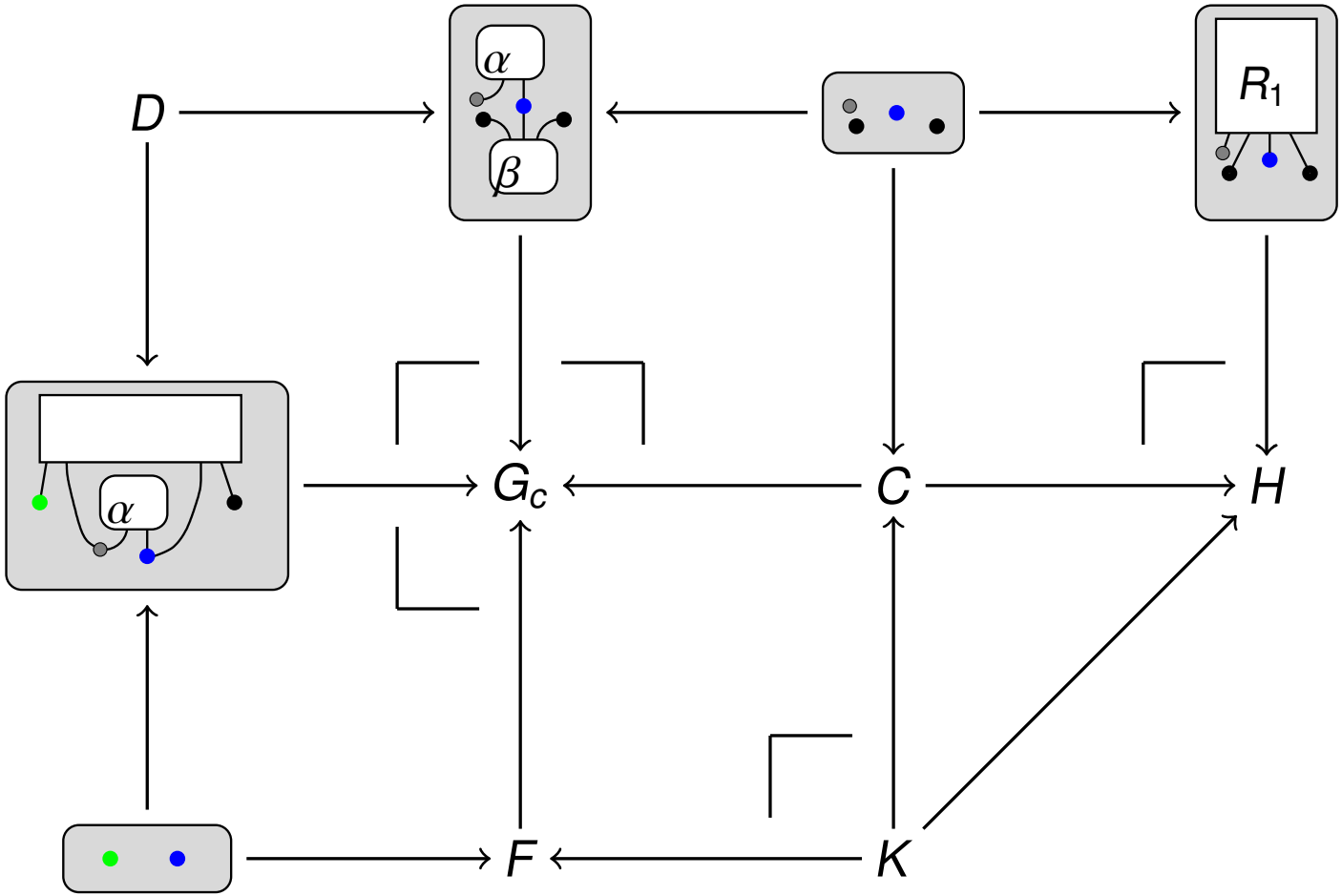
A transition



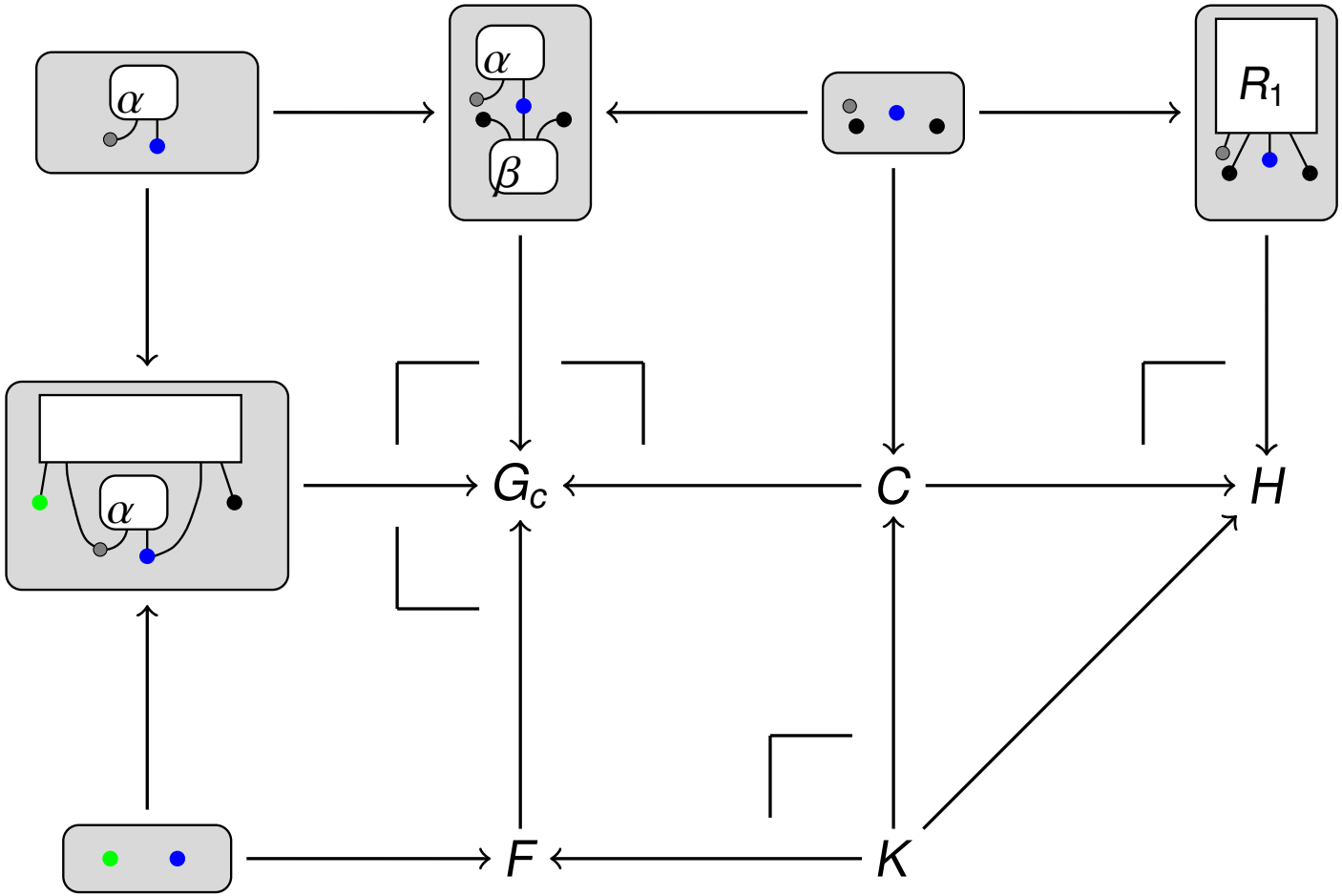
Double-PushOut with Borrowed Context



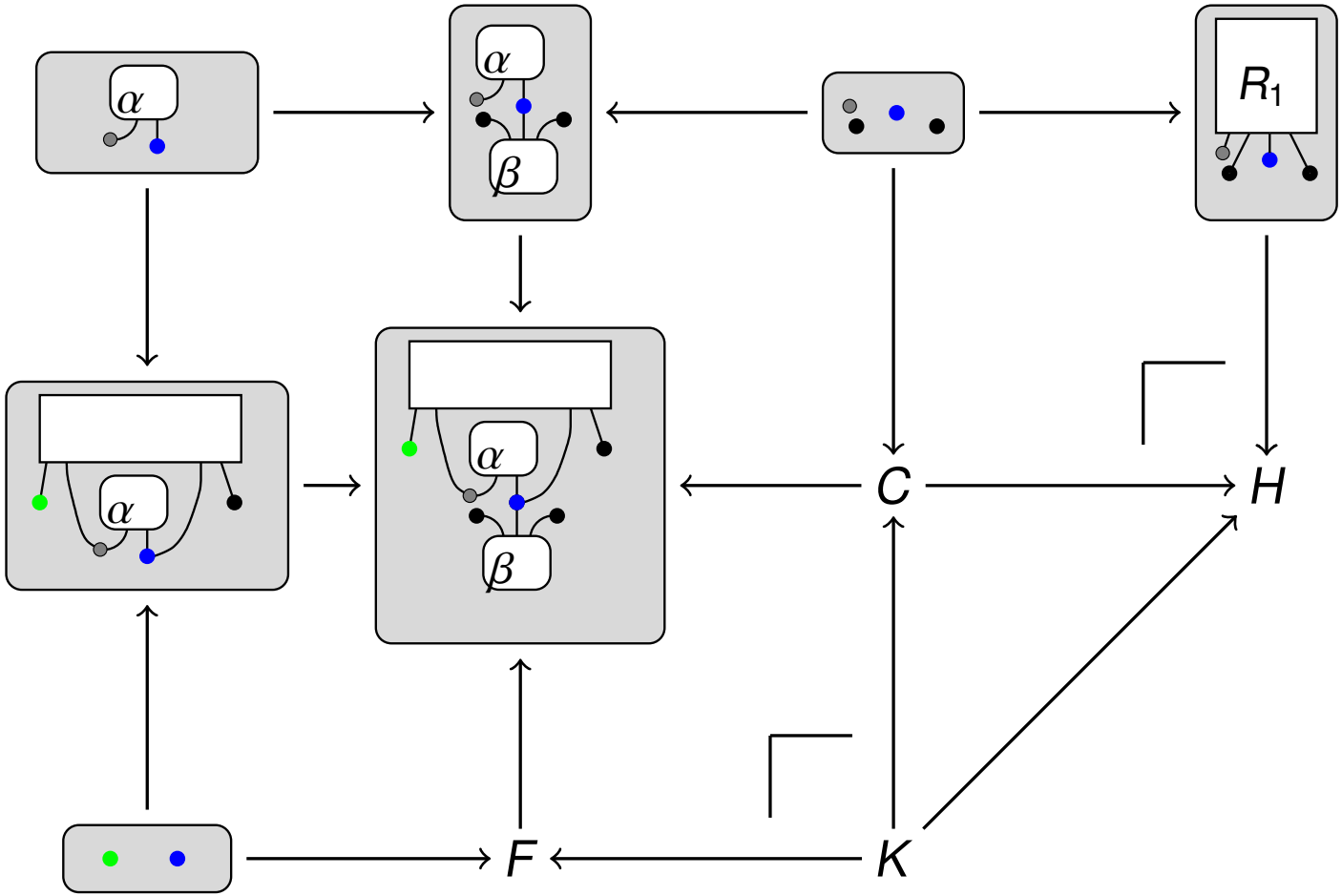
Double-PushOut with Borrowed Context



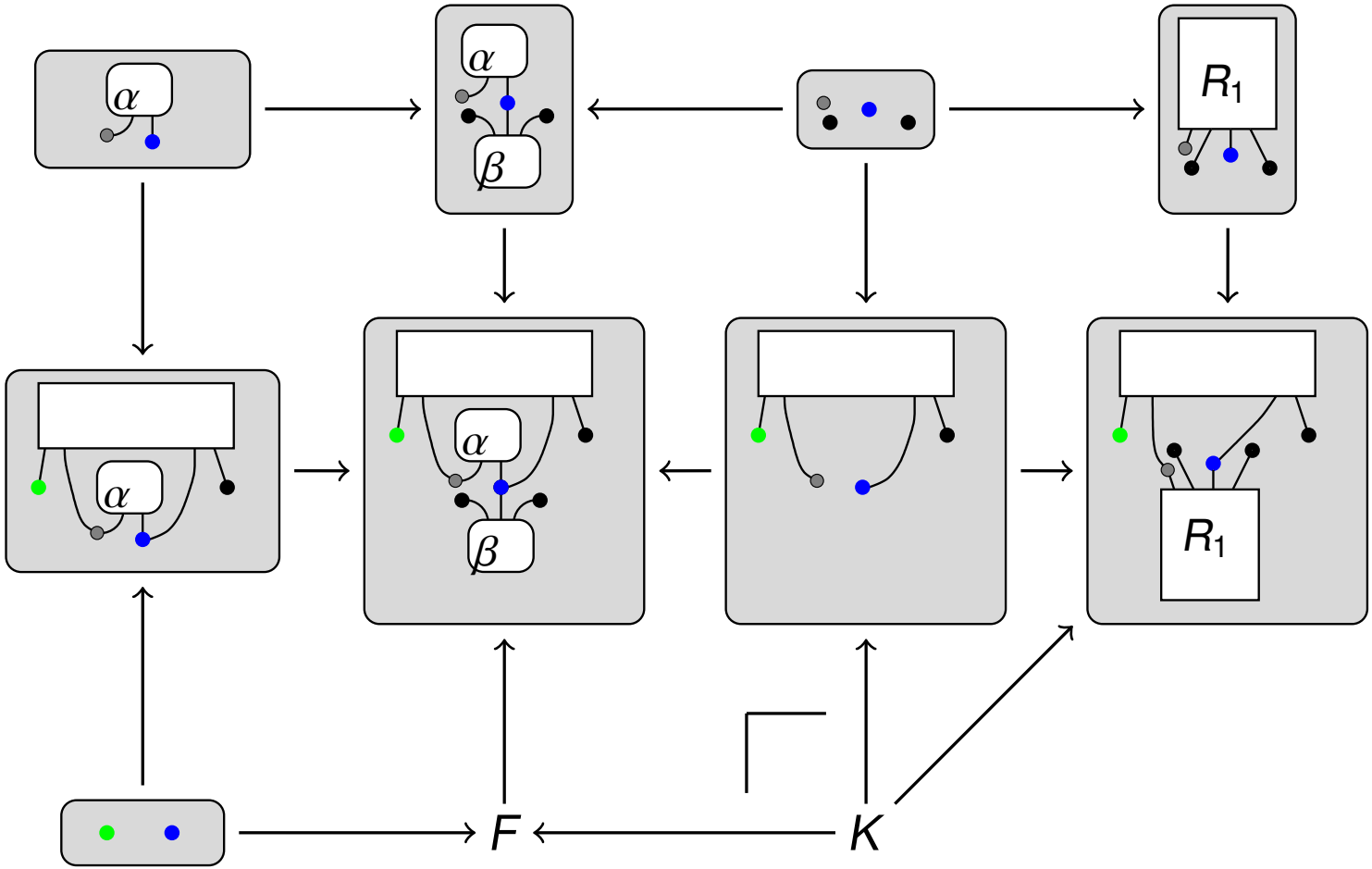
Double-PushOut with Borrowed Context



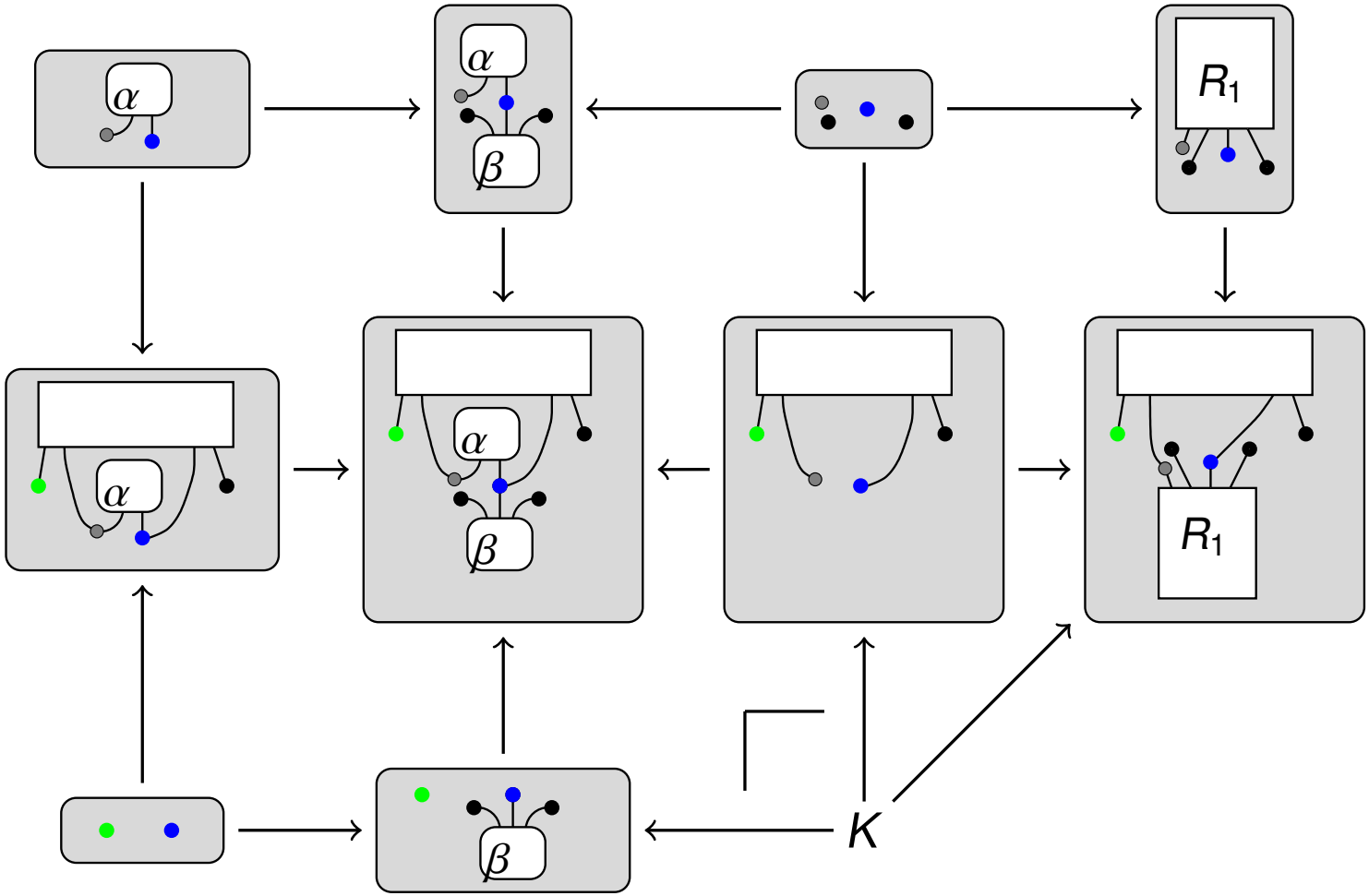
Double-PushOut with Borrowed Context



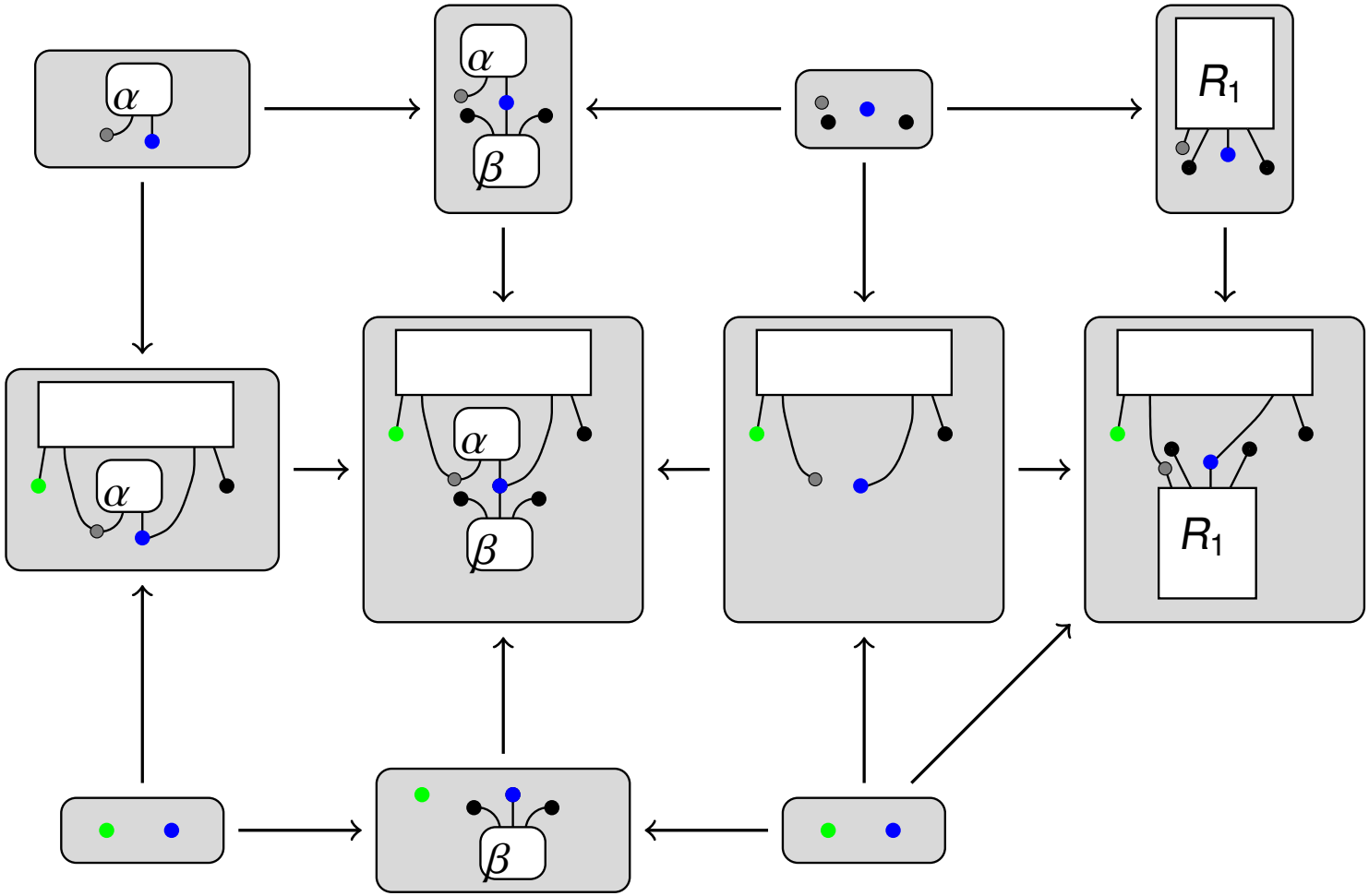
Double-PushOut with Borrowed Context



Double-PushOut with Borrowed Context

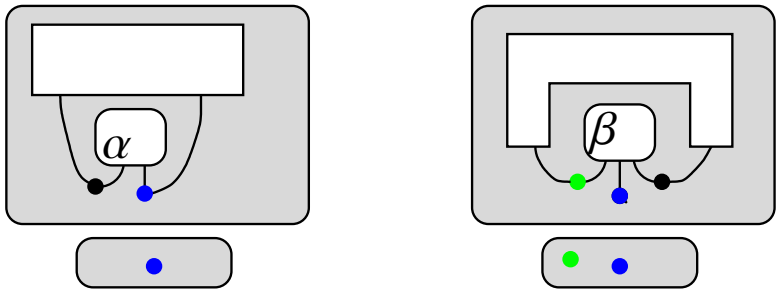


Double-PushOut with Borrowed Context



Communication

What one would want is to express some kind of “complementarity of action”:

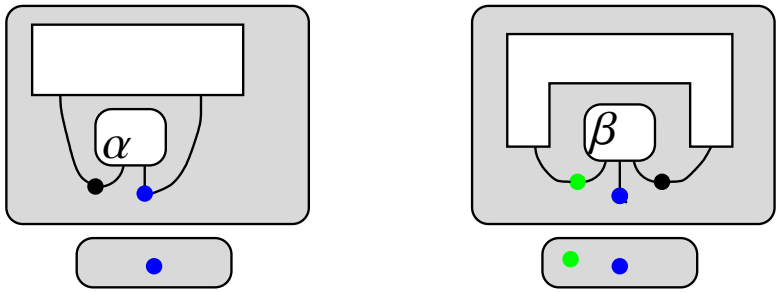


can be “put together” and yield an internal reaction (like in ccs or π-calculus...)

$$\frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\bar{a}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

Communication

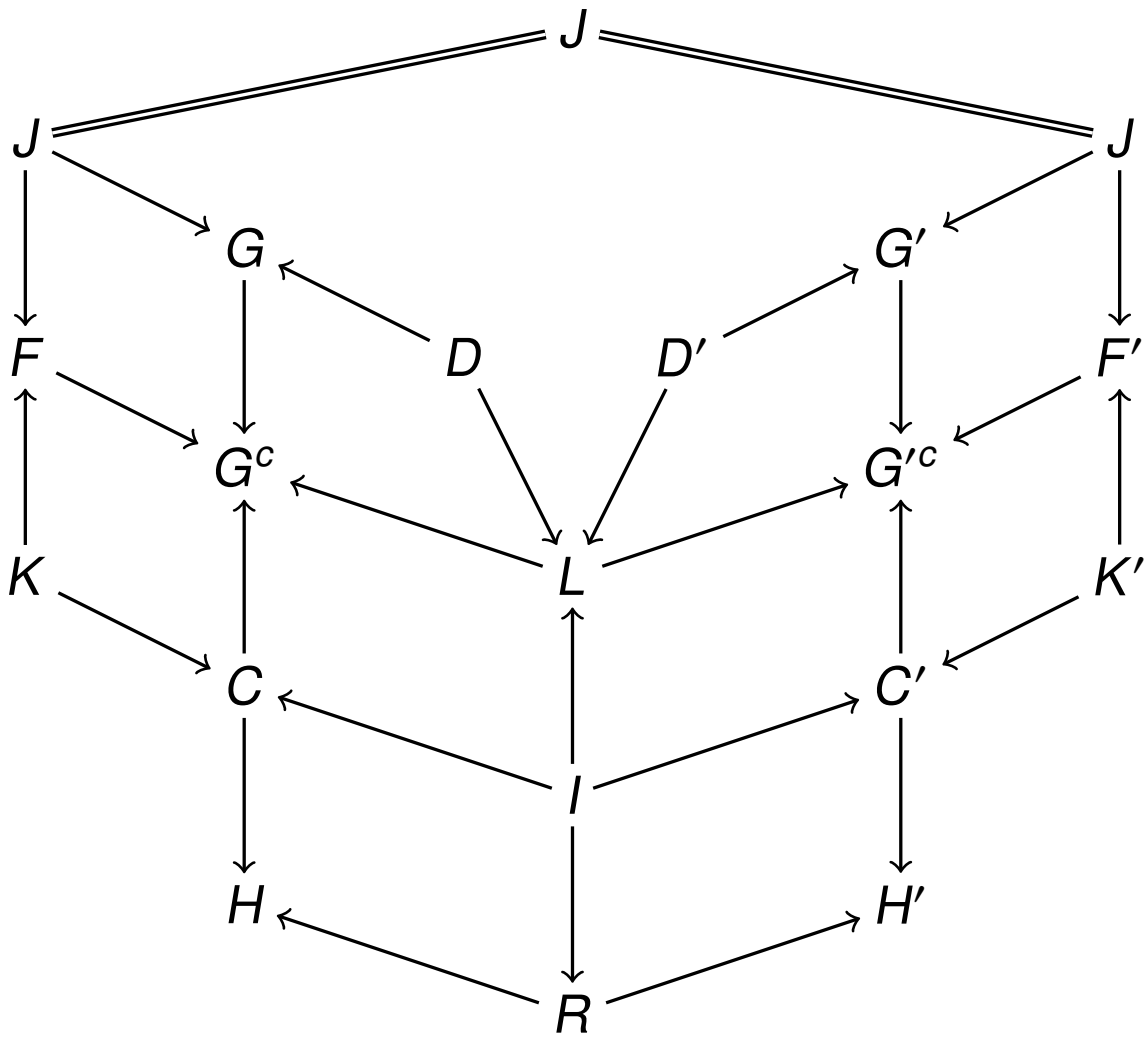
What one would want is to express some kind of “complementarity of action”:



can be “put together” and yield an internal reaction (like in ccs or π-calculus...)

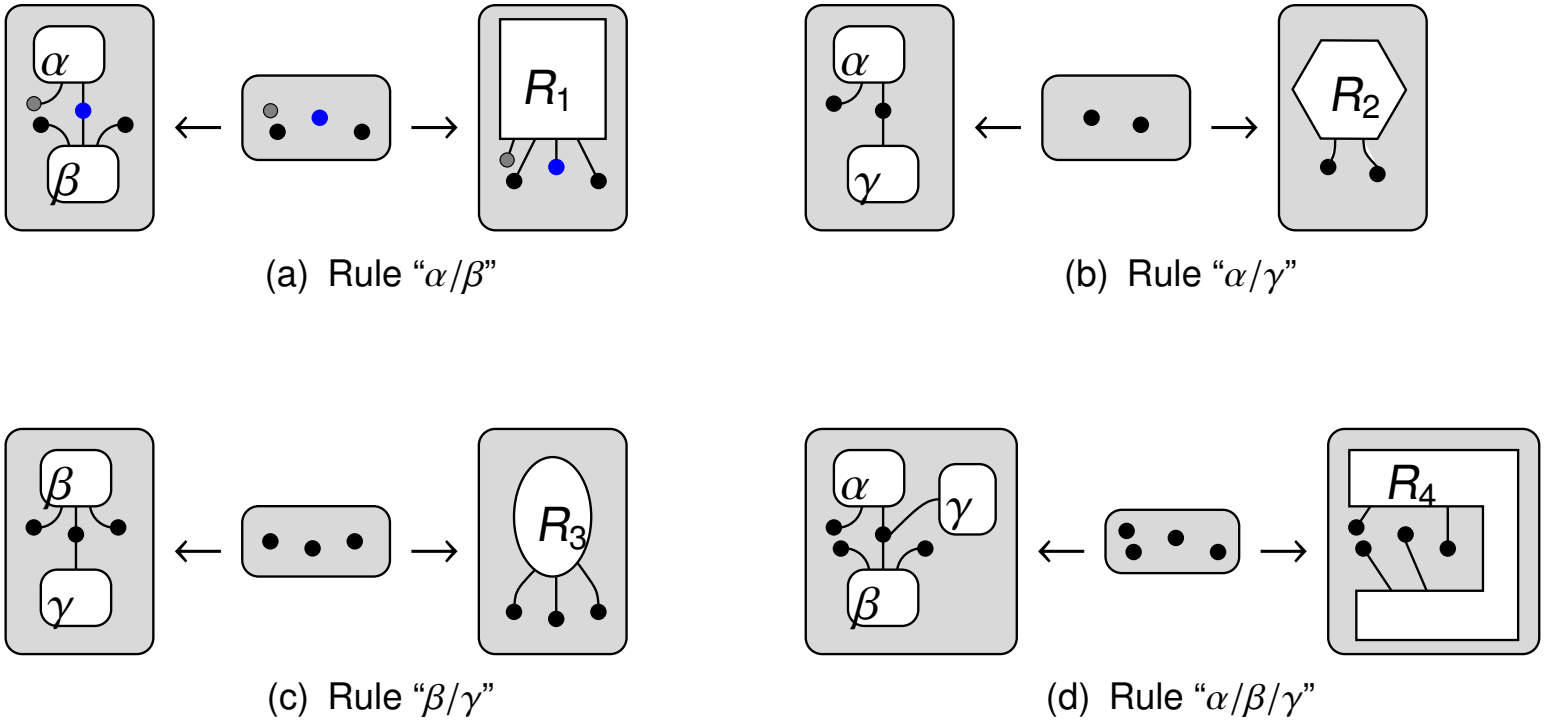
$$\frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\bar{a}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$$

Solution



Construct all with pushouts...

Graph transformation - example



(a) Rule "α/β"

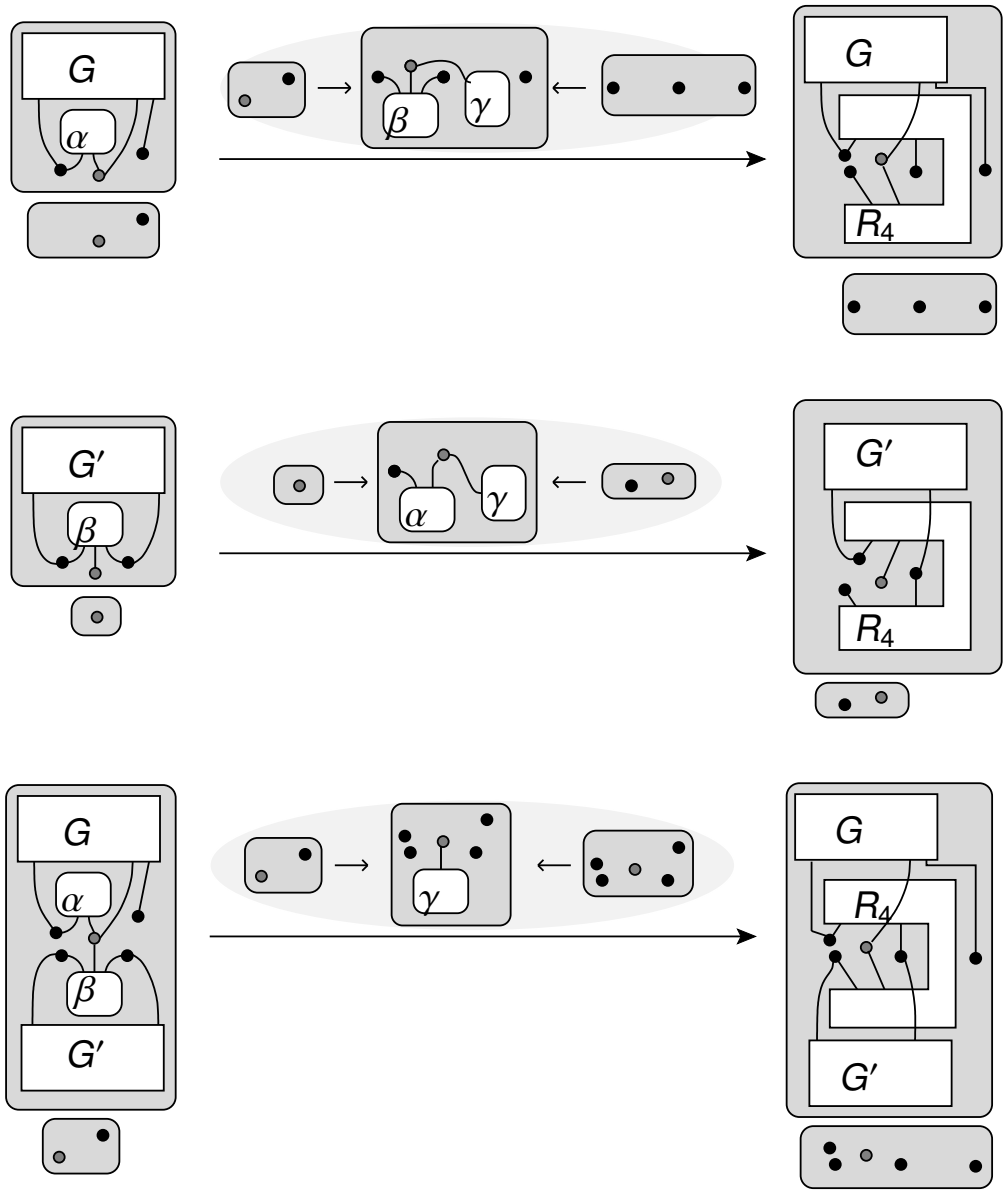
(b) Rule "α/γ"

(c) Rule "β/γ"

(d) Rule "$\alpha/\beta/\gamma$"

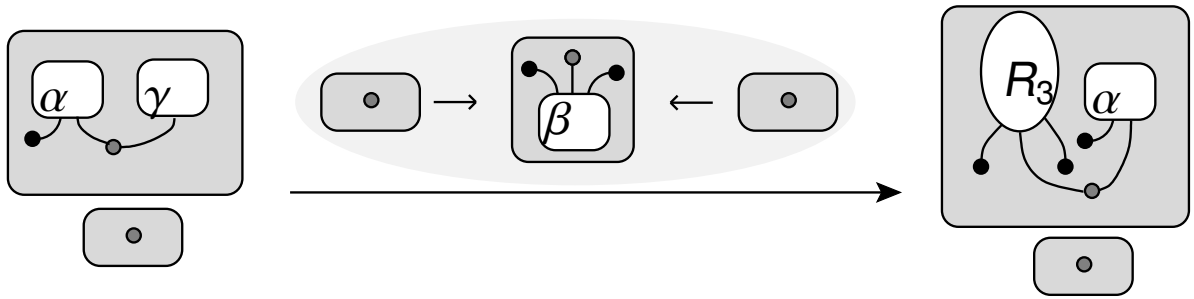
Figure: Reaction rules of a hypergraph transformation system $\mathcal{S}$.

Composition

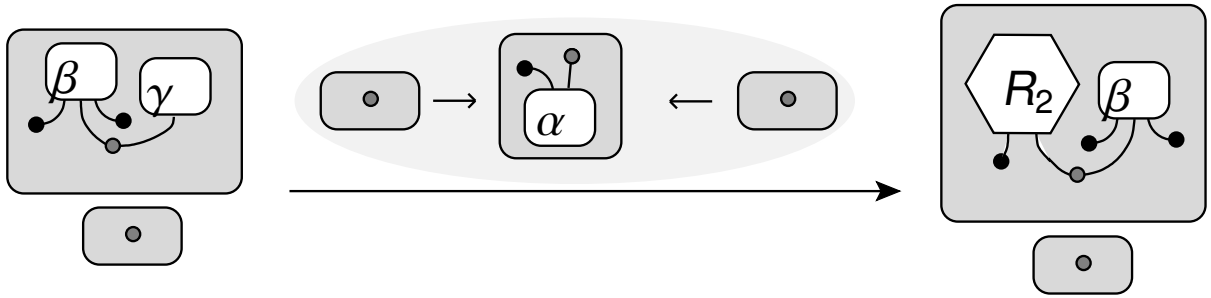


But...

It's only possible if the two rules are actually the same.



(d) A transition from rule β/γ

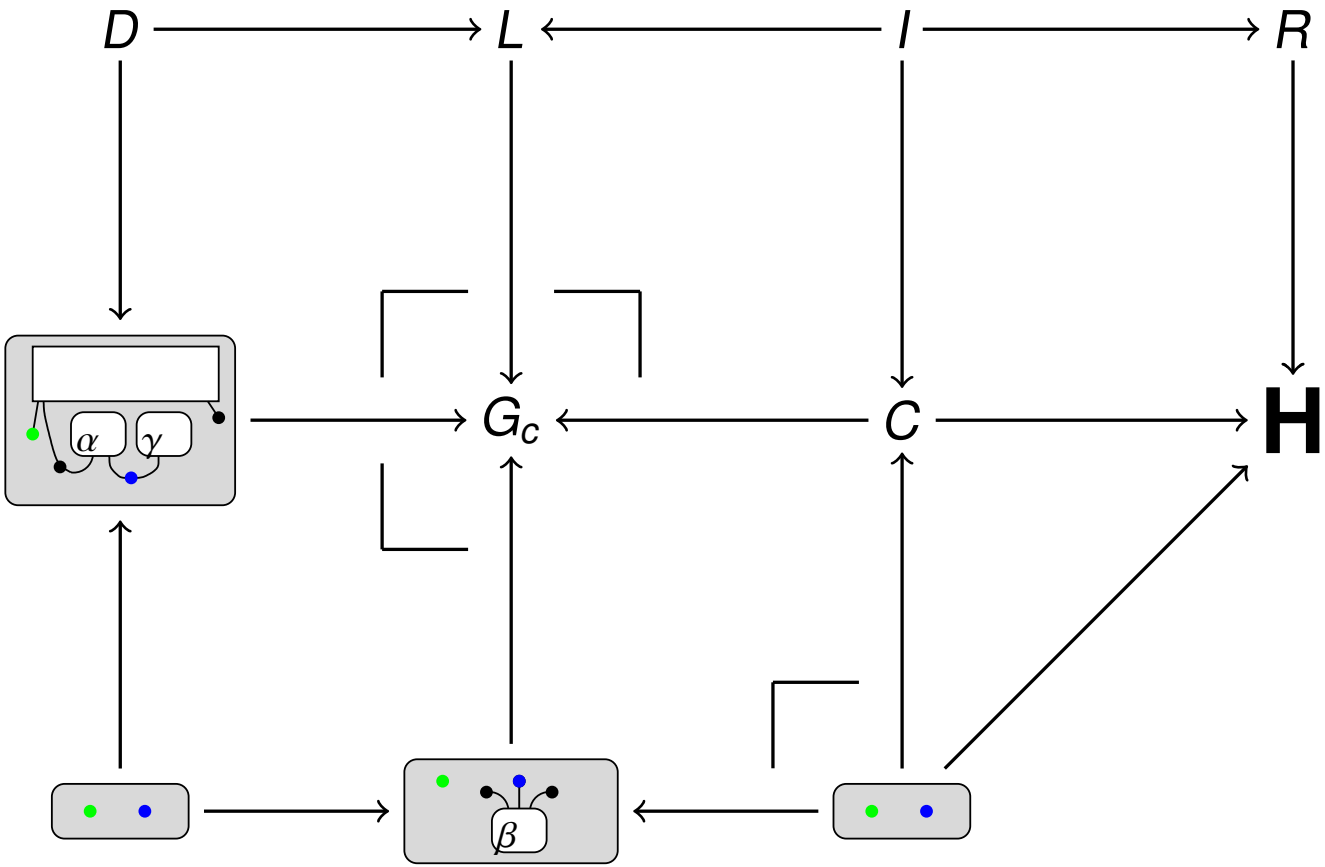


(e) A transition from rule α/γ

Figure: They seem composable...

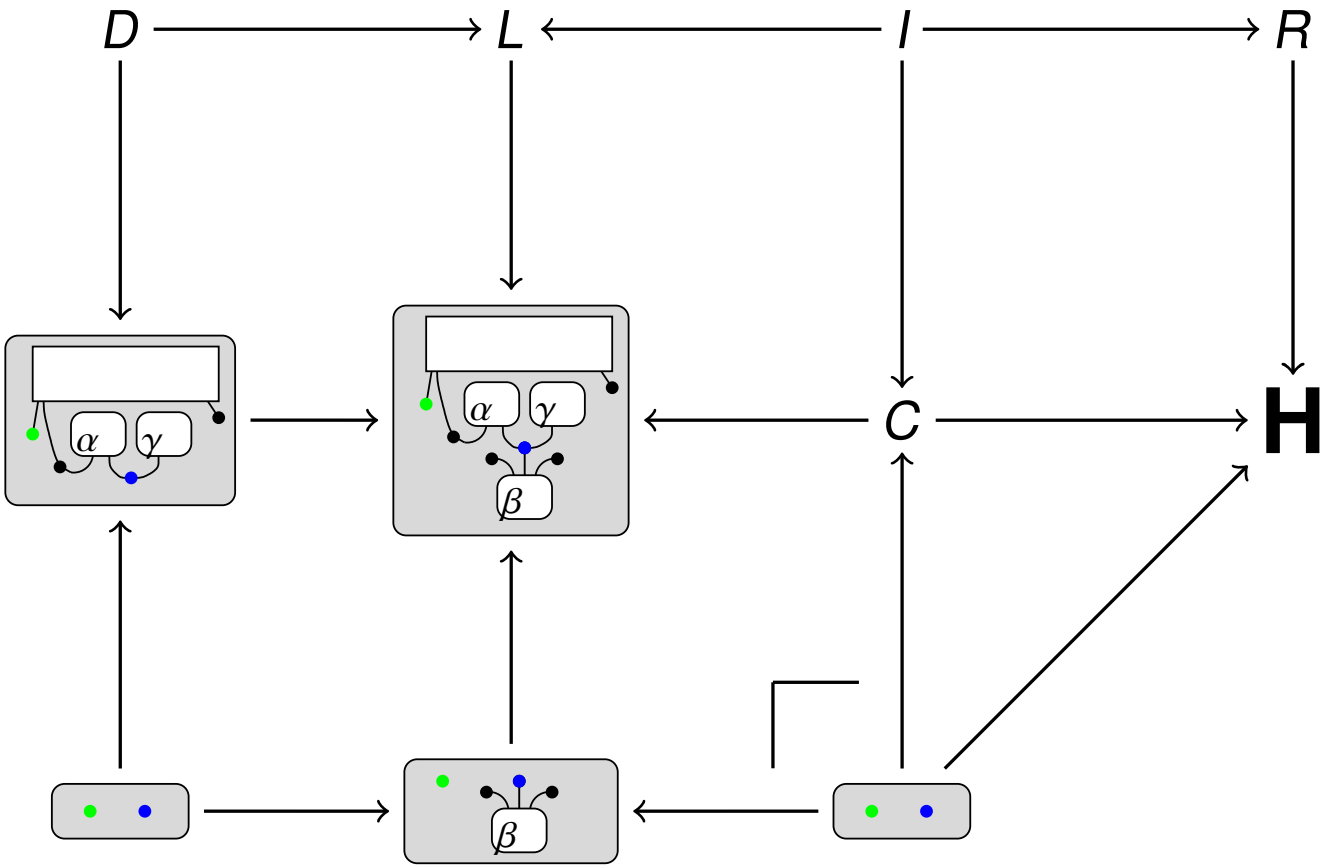
Transition uniquely determines rule

Reformation - change the rules: **unique partners**.



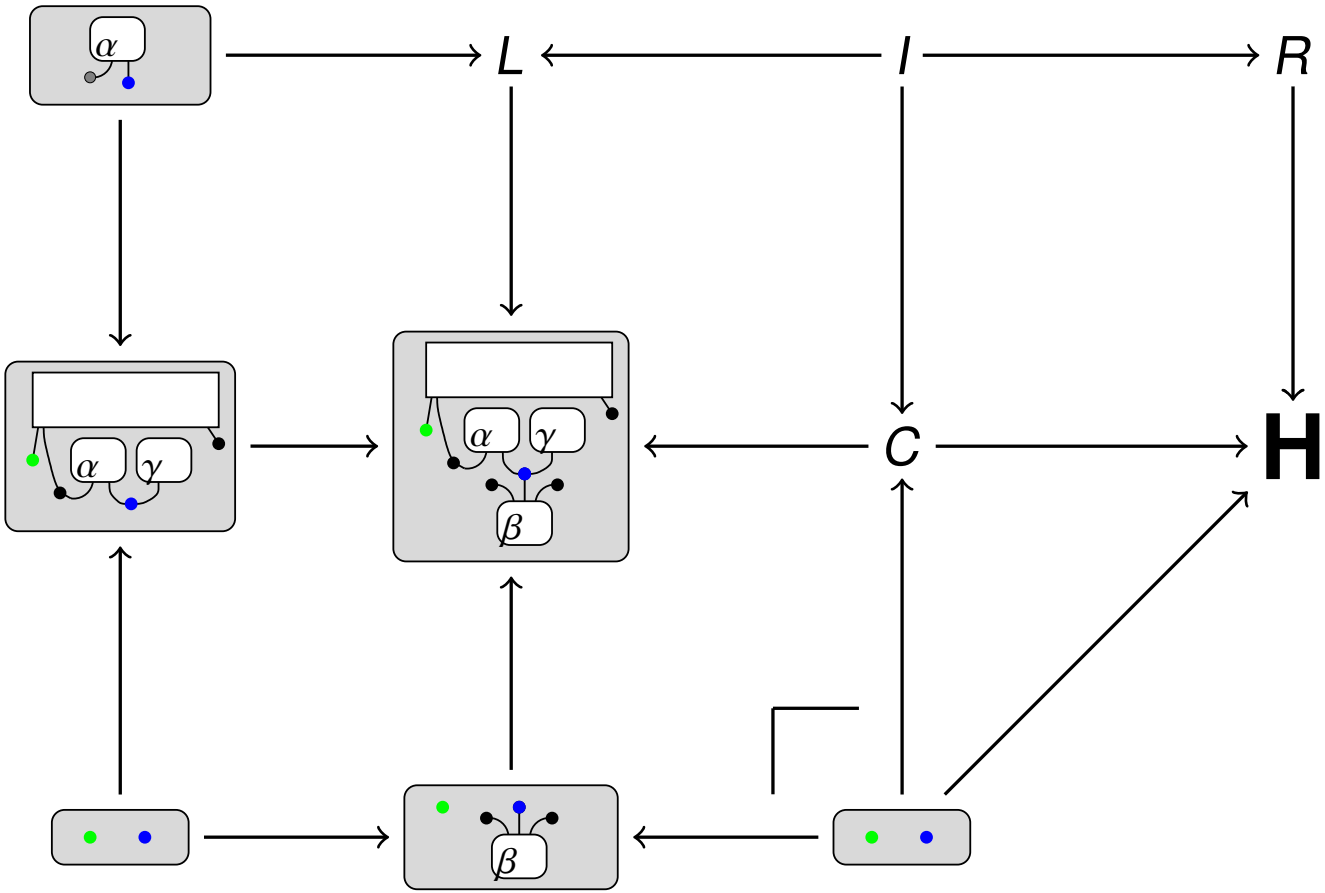
Transition uniquely determines rule

Reformation - change the rules: **unique partners**.



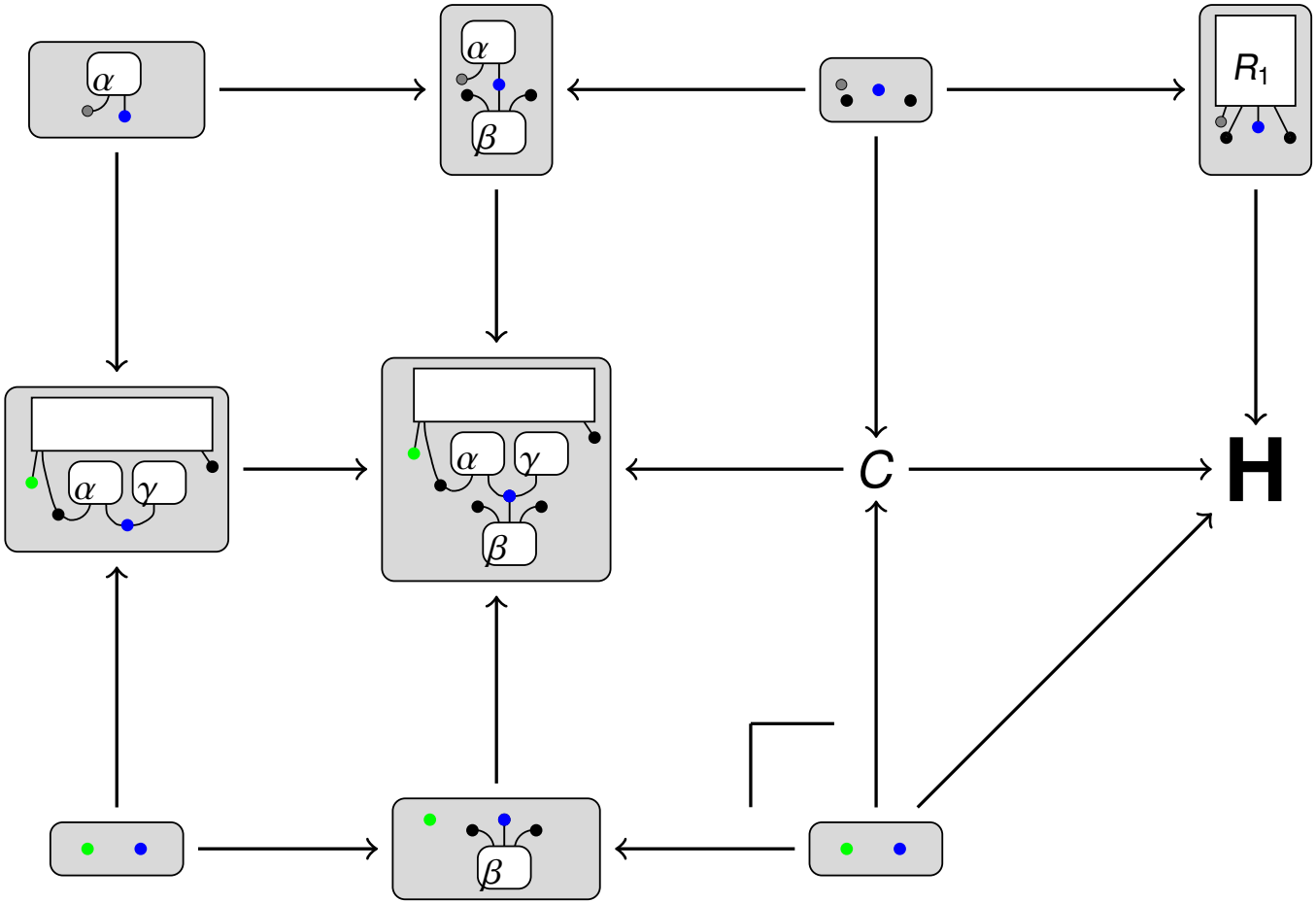
Transition uniquely determines rule

Reformation - change the rules: **unique partners**.



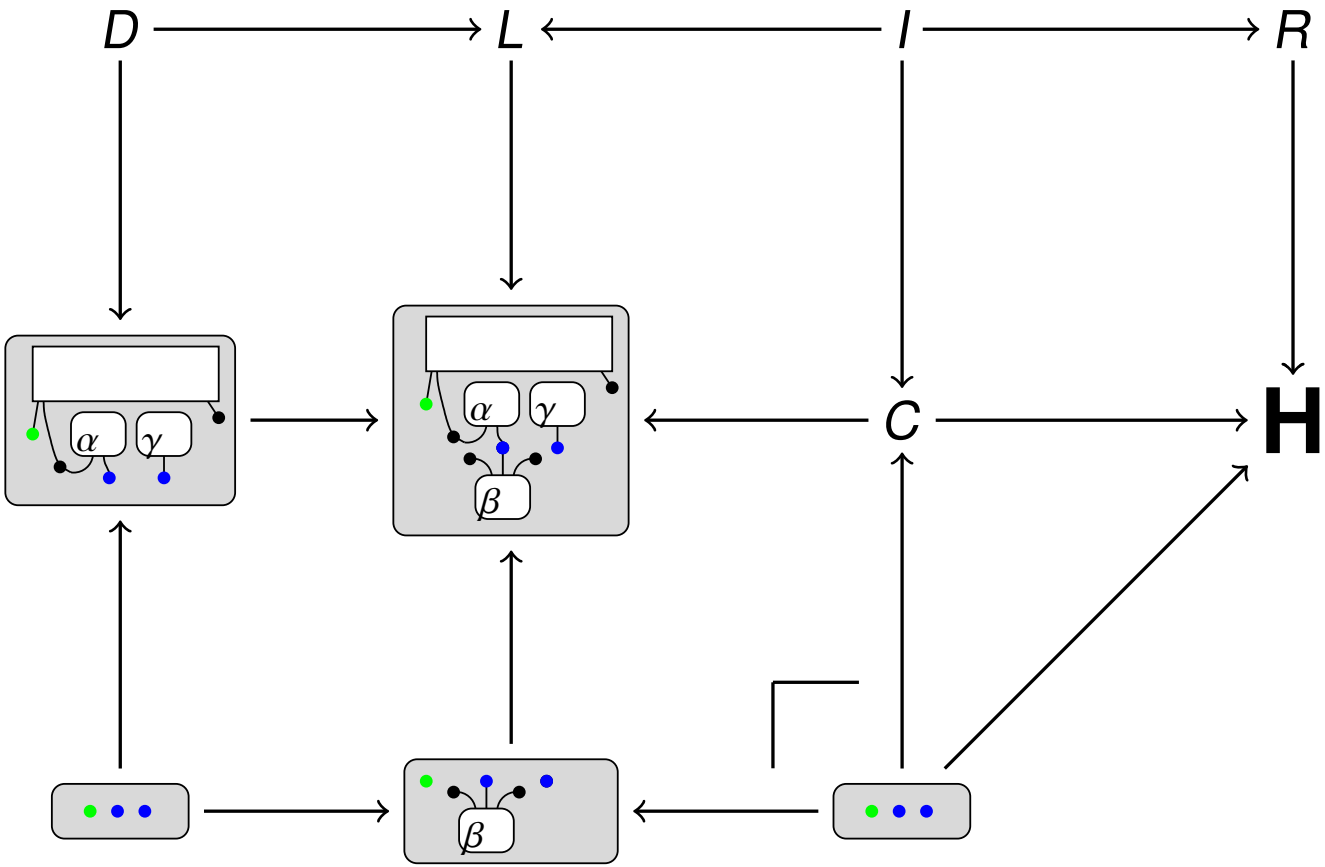
Transition uniquely determines rule

Reformation - change the rules: **unique partners**.



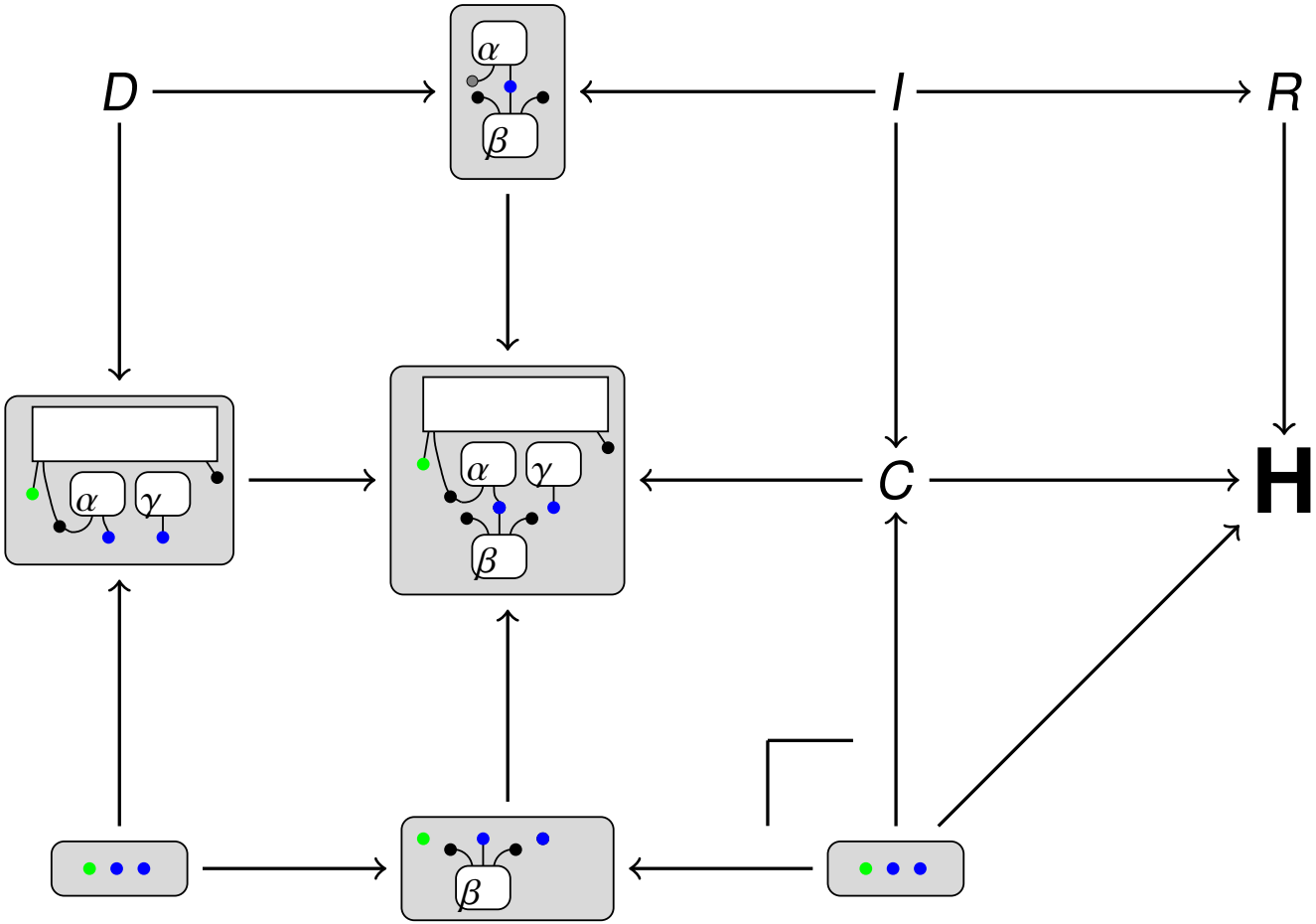
Transition uniquely determines rule

Cultural revolution - change the language: **simple wires**.



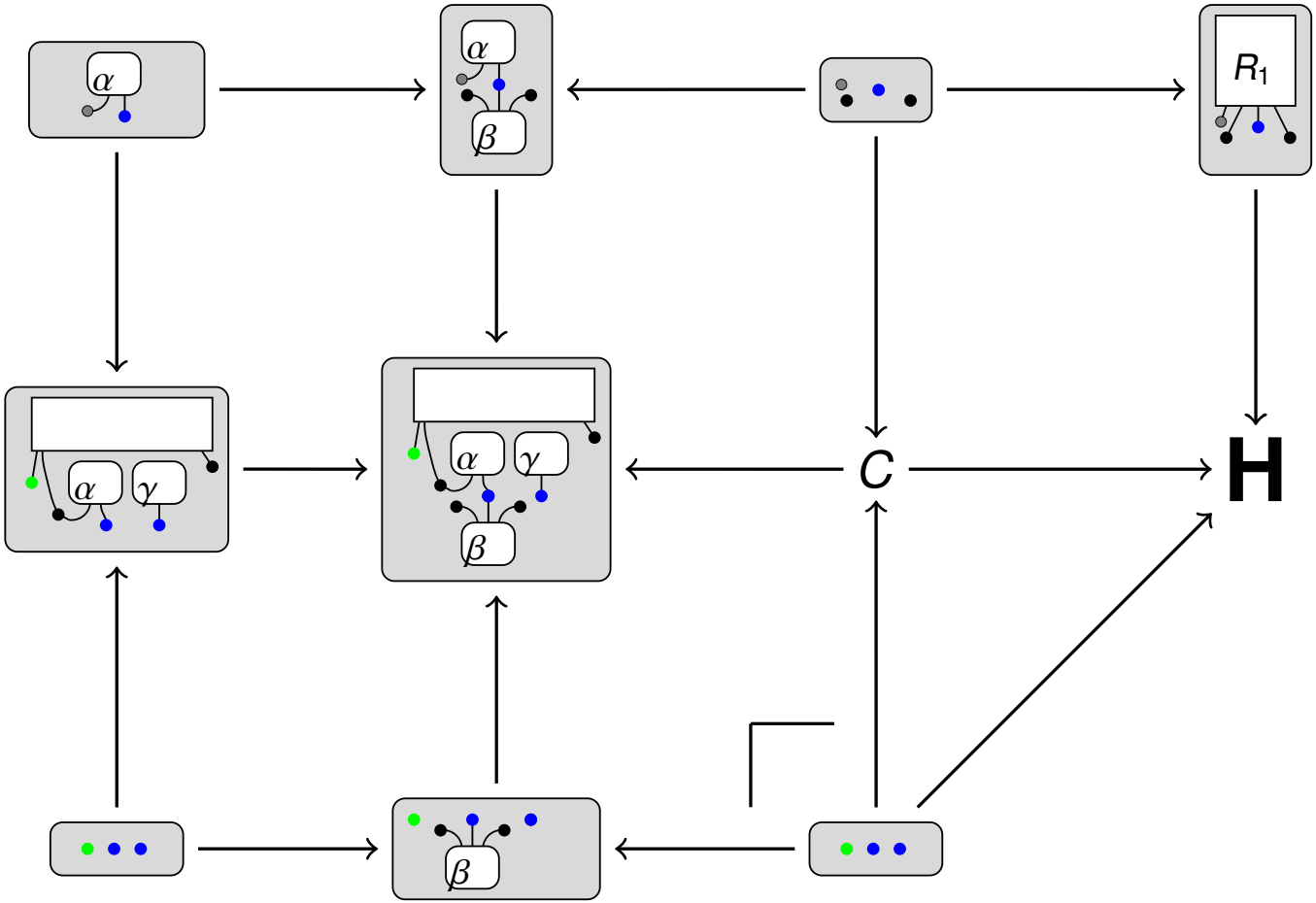
Transition uniquely determines rule

Cultural revolution - change the language: **simple wires**.



Transition uniquely determines rule

Cultural revolution - change the language: **simple wires**.



Further work

- Generalize the condition (categorically perhaps).
- More general morphisms (especially for interface).
- Composing transitions without rebuilding the whole graph.

Thank you.