Compositional Graph Pattern Matching

Malin Altenmüller University of Edinburgh

jww Ross Duncan

ACT 2026 Tallinn

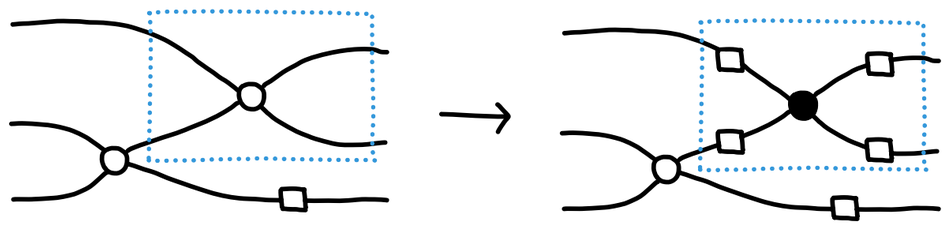
Why graphs?

- combinatorial representation of string diagrams

- graphical properties correspond to coherence laws

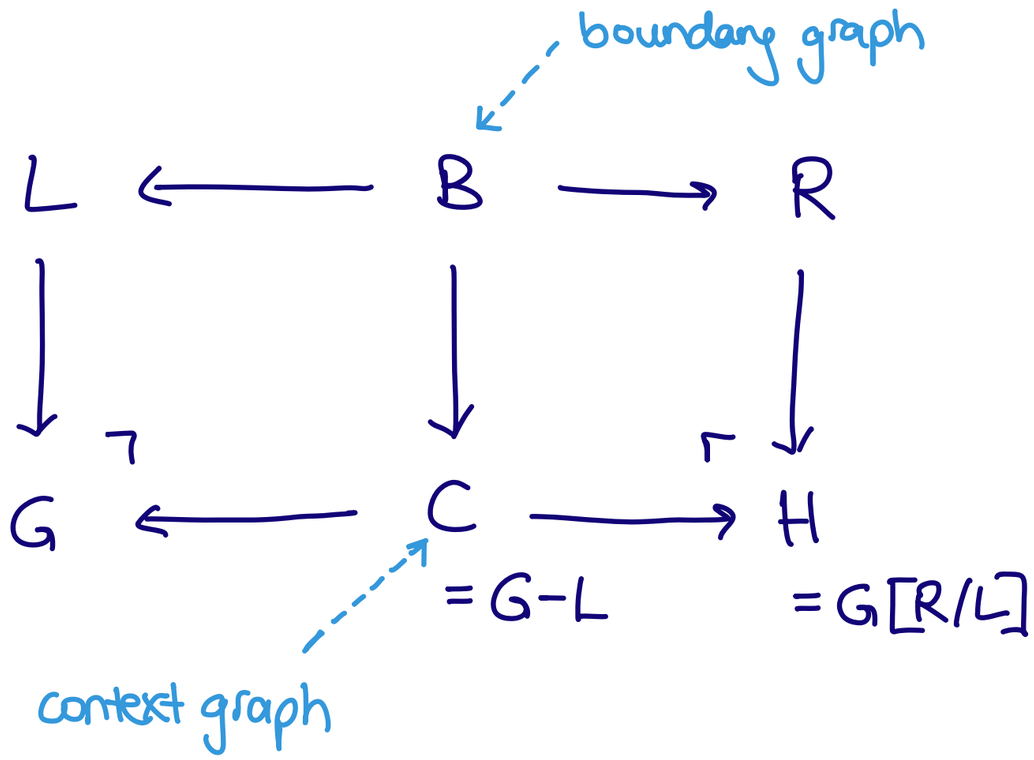
- equational theory → rewrite rules, implemented as graph rewriting

Example: colour change rule in ZX-calculus



How to implement graphs and their rewriting?

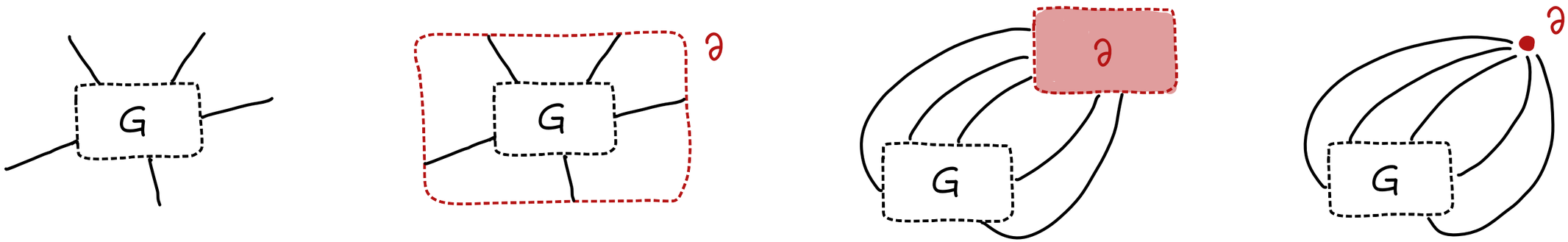
Double-Pushout (DPO) rewriting



- required properties:
  1. pushouts exist
  2. pushout complements exist & are unique.

Surface-embedded graphs - interface edges

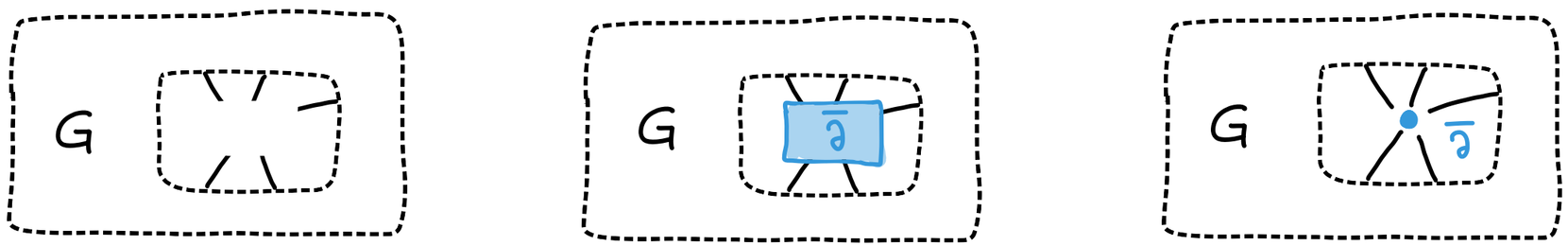
- introduce a boundary vertex to capture interface edges



- explicit element for an "empty region" in the graph

Surface-embedded graphs - holes

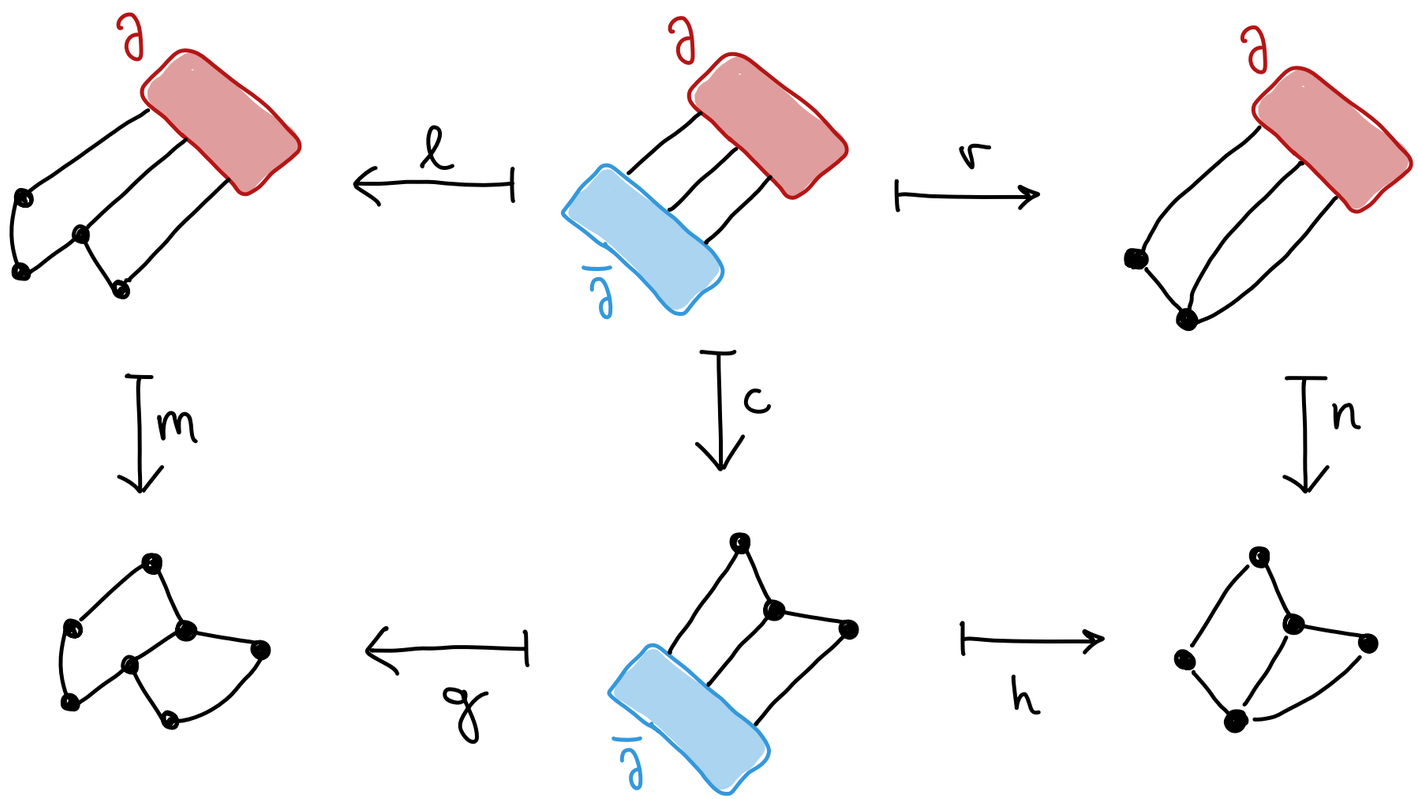
- dual boundary vertices stand for any hole in a graph



- subtleties in the development of a suitable category
  (morphisms have to be partial)

- category of surface-embedded graphs has pushouts & pushout complements*

Double-Pushout (DPO) Rewriting – Example

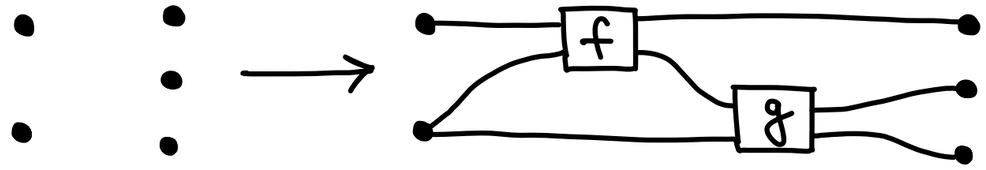


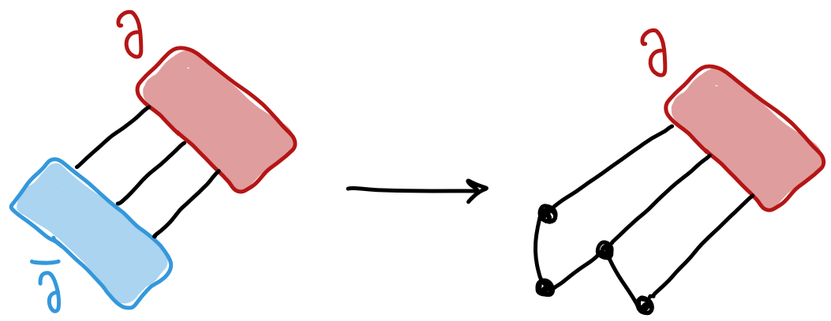
Observation: only one region is instantiated at a time

Disclaimer

I will continue to use surface-embedded graphs but the following construction works for categories in which

- rewriting by double-pushout is well defined
- have a notion of boundary and boundary graph





Graphs as Multicategories*

*aka Coloured Operads

cf. Spivak's Operad of Wiring Diagrams and the Little Discs Operad (e.g. Leinster)

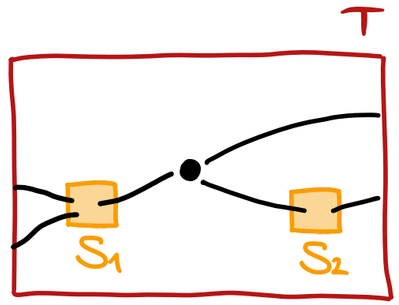
- objects R, S, T are boundaries of graphs

"variables"

- maps $S_1, S_2, \dots, S_n \to T$ are graphs with

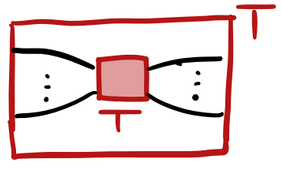
  > holes of types $S_1, S_2, \dots, S_n$

  > interface of type T



$G : S_1, S_2 \to T$

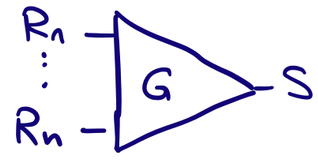
- identity $T \to T$ is the boundary graph on T

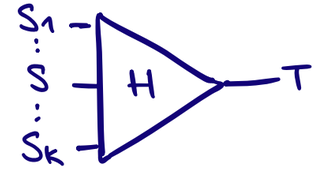


<u>Observation:</u> maps specify the structure in between variables

Multicategory composition

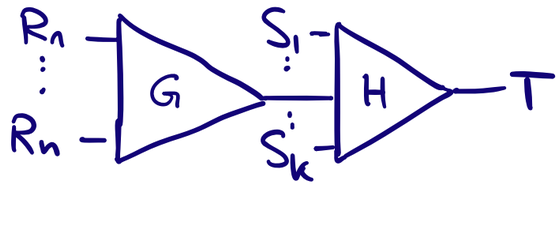
Given $G : R_1, \dots, R_n \to S$ and $H : S_1, \dots, S_i, \dots, S_k \to T$

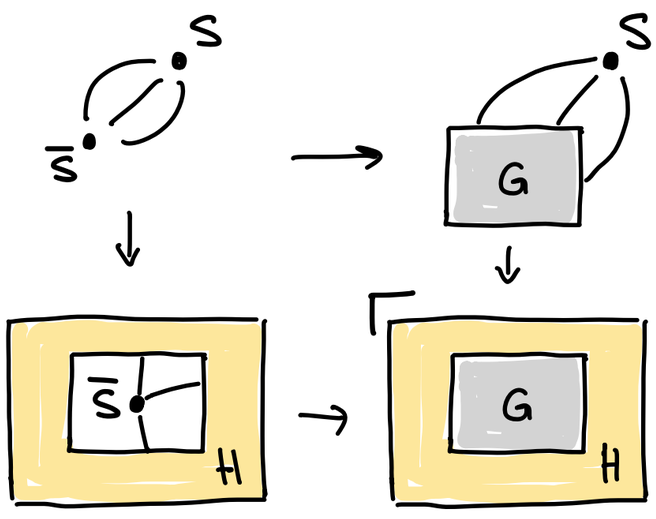




calculate $G;H : S_1, \dots, R_1, \dots, R_n, \dots, S_k \to T$

by <u>substituting</u> G into the hole S in H



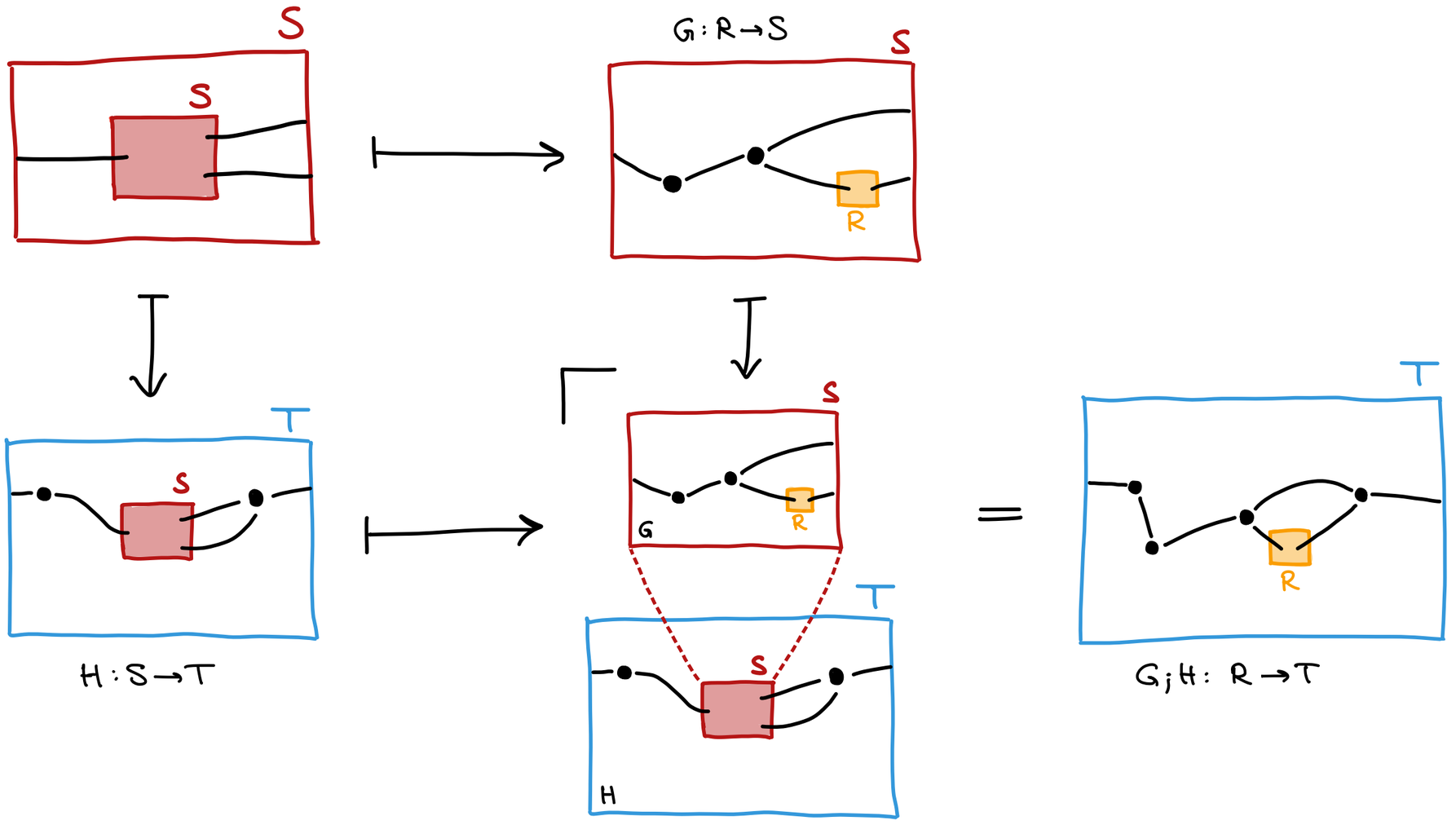


implemented as a pushout along S

in the underlying graph category

Multicategory composition - Example

Special case
one input - one output



Pattern matching in functional programming languages (1)

- programming technique for specifying functions over inductive data

Example: lists

```
List a = Nil | a : List a

list :: List Int
list = 13 : 24 : 8 : 100 : Nil
```

terms: how to build data

function on (numeric) lists:

```
sum :: List Int → Int
sum Nil = 0
sum (x : xs) = x + sum xs
```

pattern variables

patterns: how to take data apart

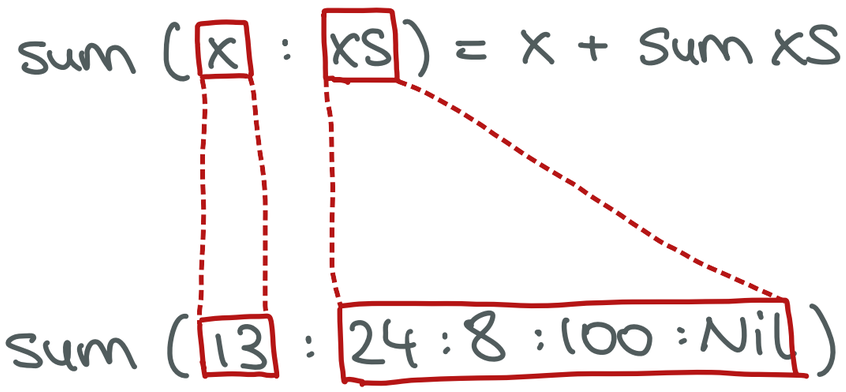
Pattern matching in functional programming languages (2)

```
List a = Nil | a : List a

list :: List Int
list = 13 : 24 : 8 : 100 : Nil
```

```
sum :: List Int → Int
sum Nil = 0
sum (x : xs) = x + sum xs
```

what happens when we call sum list?



1. matching data against pattern
2. pattern variables are instanciated (to be used on the RHS)

Co-multicategory of graph patterns

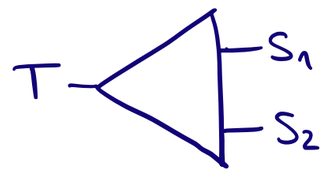
- objects R, S, T are boundaries of graphs

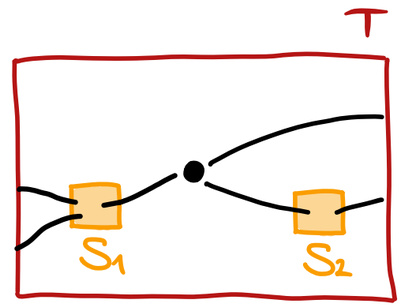
"pattern variables"

- maps $T \to S_1, S_2, \dots, S_n$ are graphs with

  > holes of types $S_1, S_2, \dots, S_n$

  > interface of type T





P: T → S1, S2

- composition by substitution "refinement"

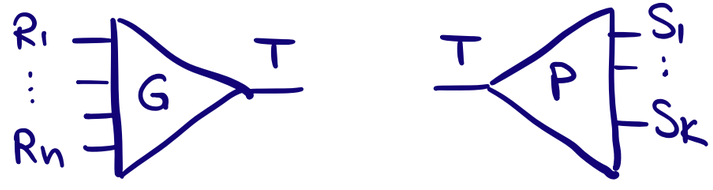
- identity $T \to T$ is the wildcard pattern

How do graphs and patterns interact?

Matching a graph against a pattern

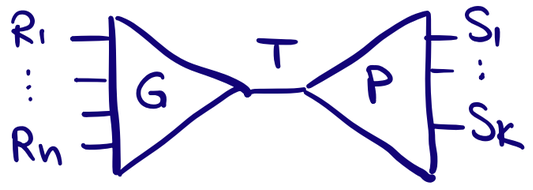
Given a graph $G : R_1, \dots, R_n \to T$

and a pattern $P : T \to S_1, \dots, S_K$,

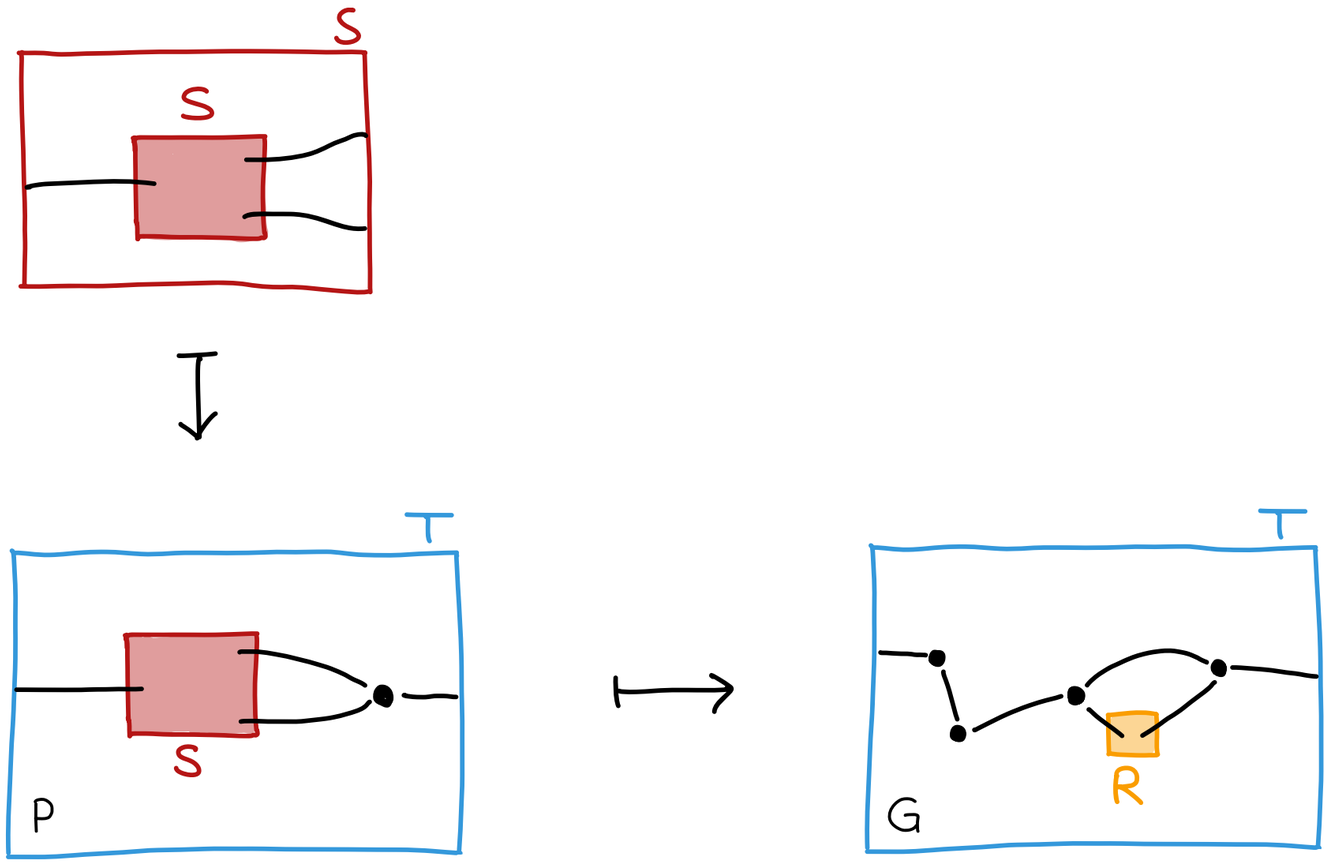


the <u>match of G against P</u> is a composite in the category of graphs

$$m : B_S \to P \to G$$

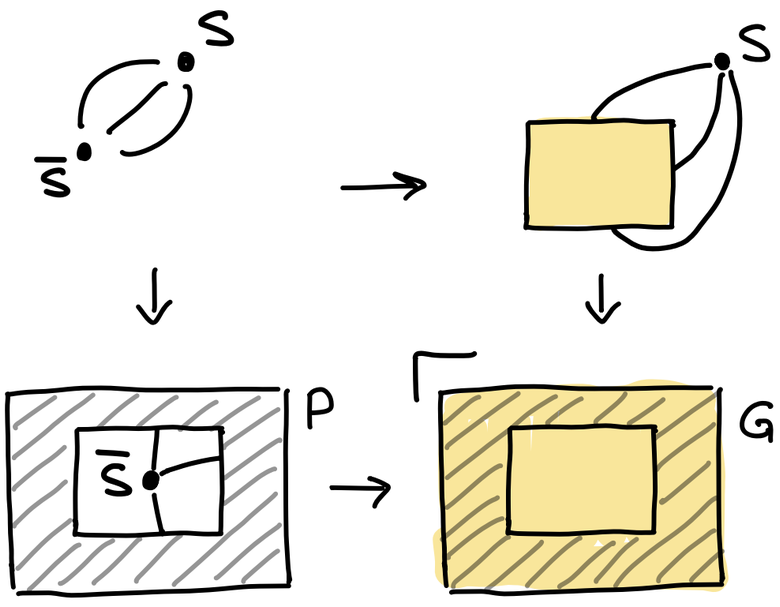


Matching a graph against a pattern - Example

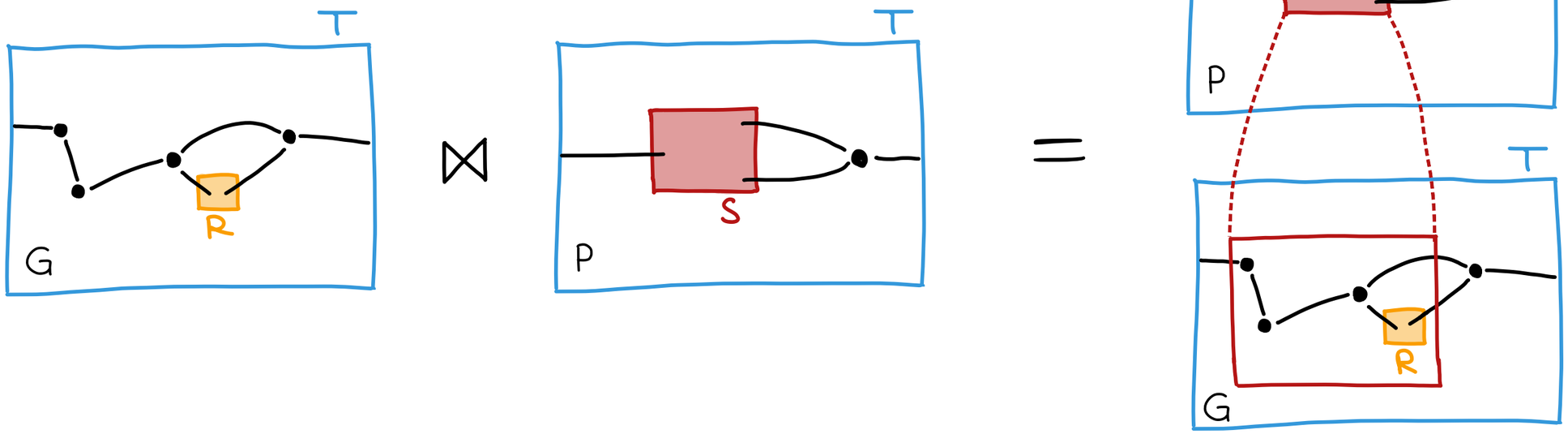


Calculating the result of a match

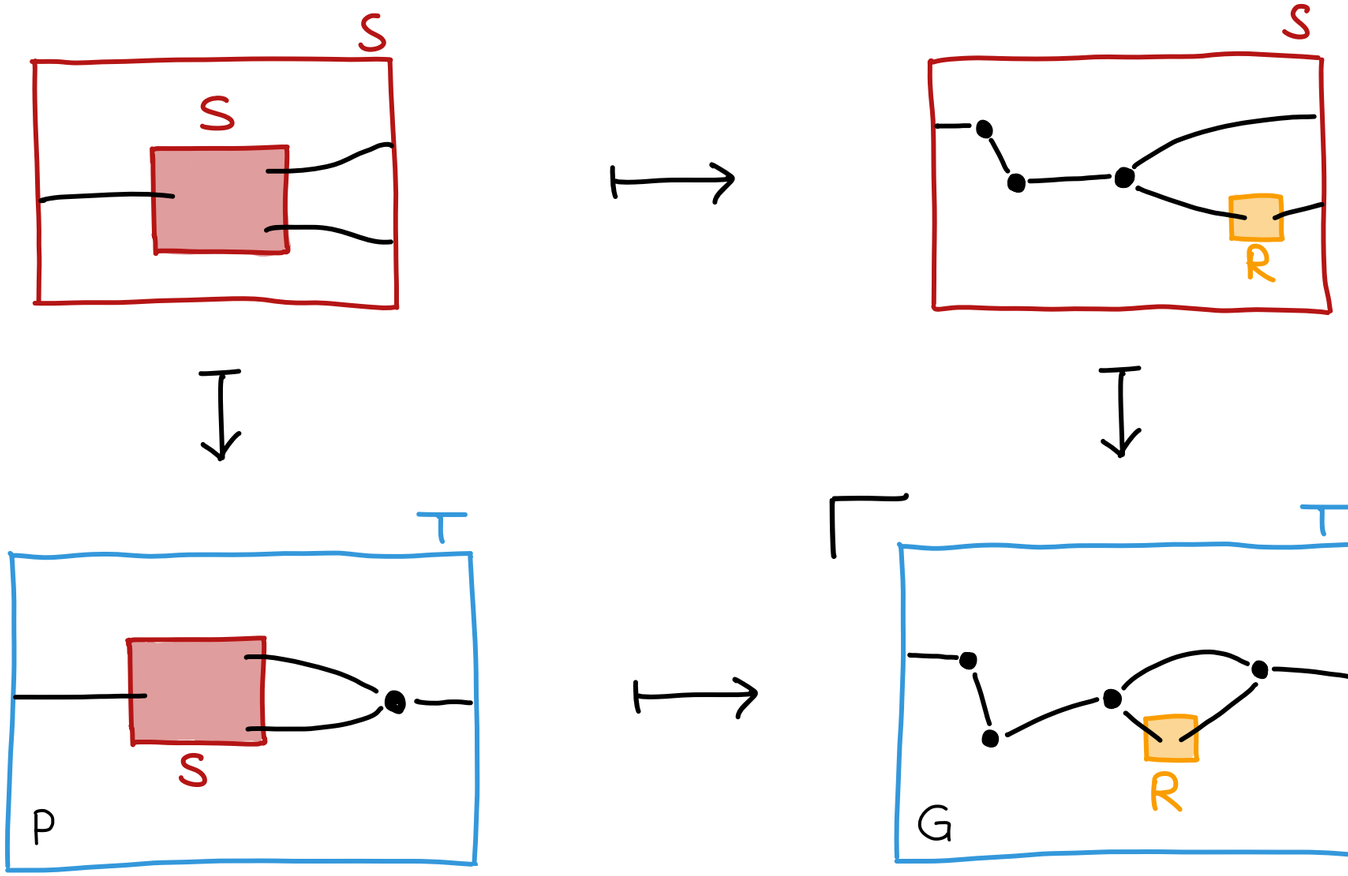
The result of a pattern match $m : B_S \to P \to G$ is calculated as its pushout complement



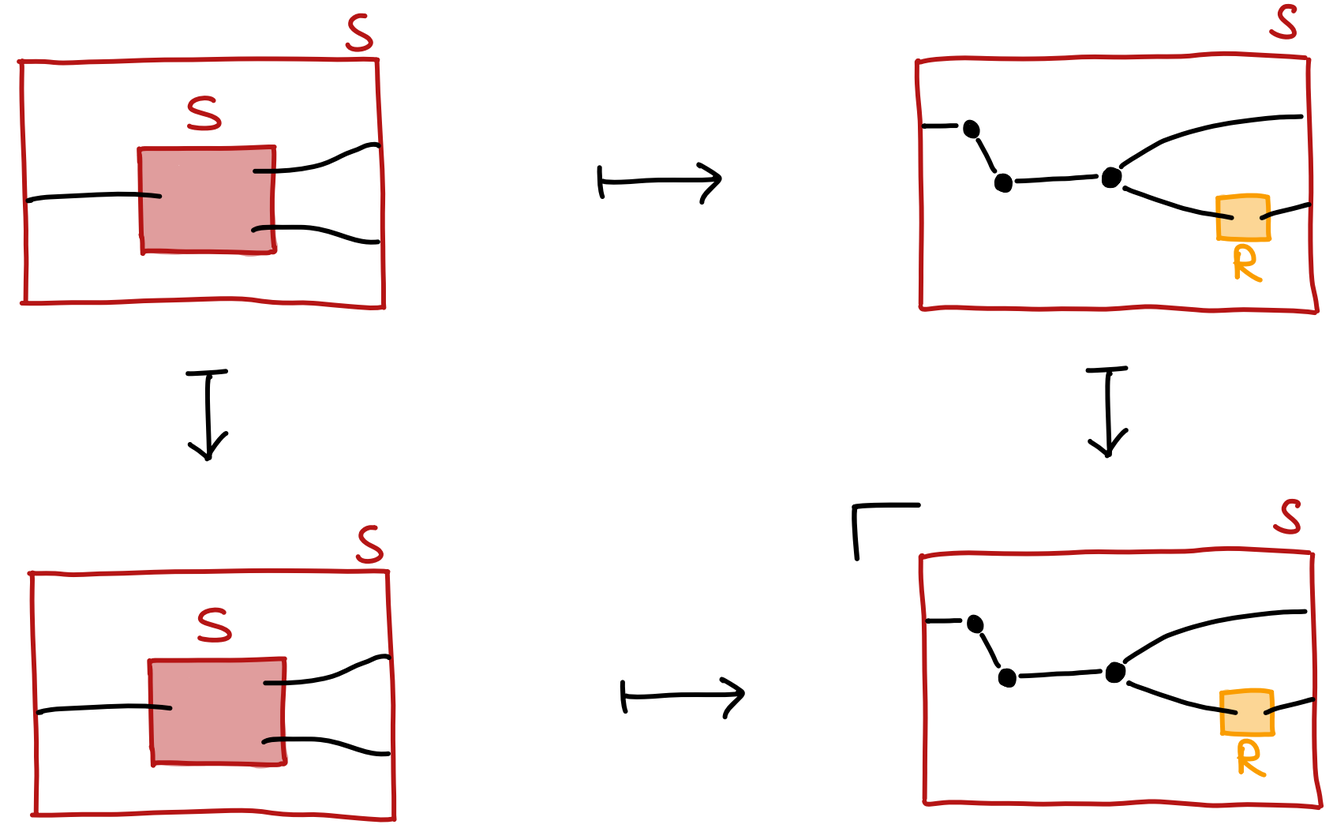
Calculating the result of a match - Example (1)



Calculating the result of a match – Example (2)



Matching against the wildcard pattern



wildcard pattern

is matched by any graph

Summary

- Combine double-pushout rewriting
  multicategory of graphs
  functional programming to get pattern matching for graphs
- Implement rewriting via polymorphic graph rewrite rules

Thank you for your attention!

malin.altenmuller@ed.ac.uk
maltenmuller.github.io