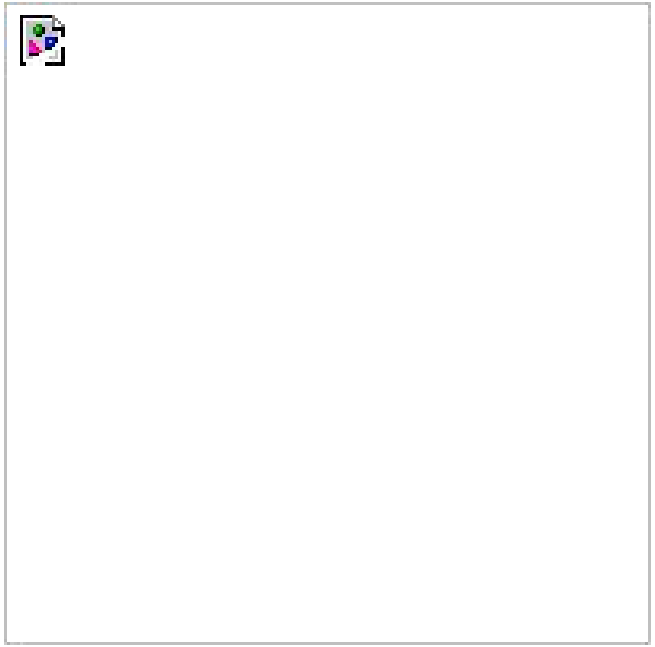
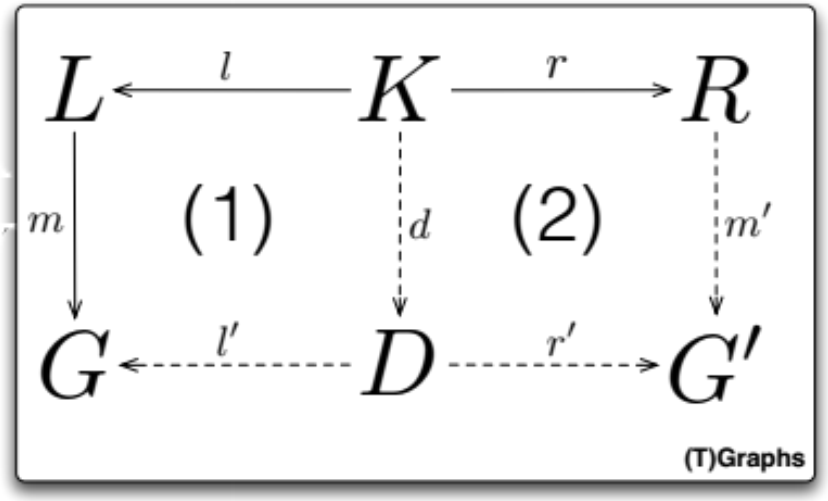
Single Pushout Approach

Patrick Steffens
&
Daniel Tigges

Running example: Pacman

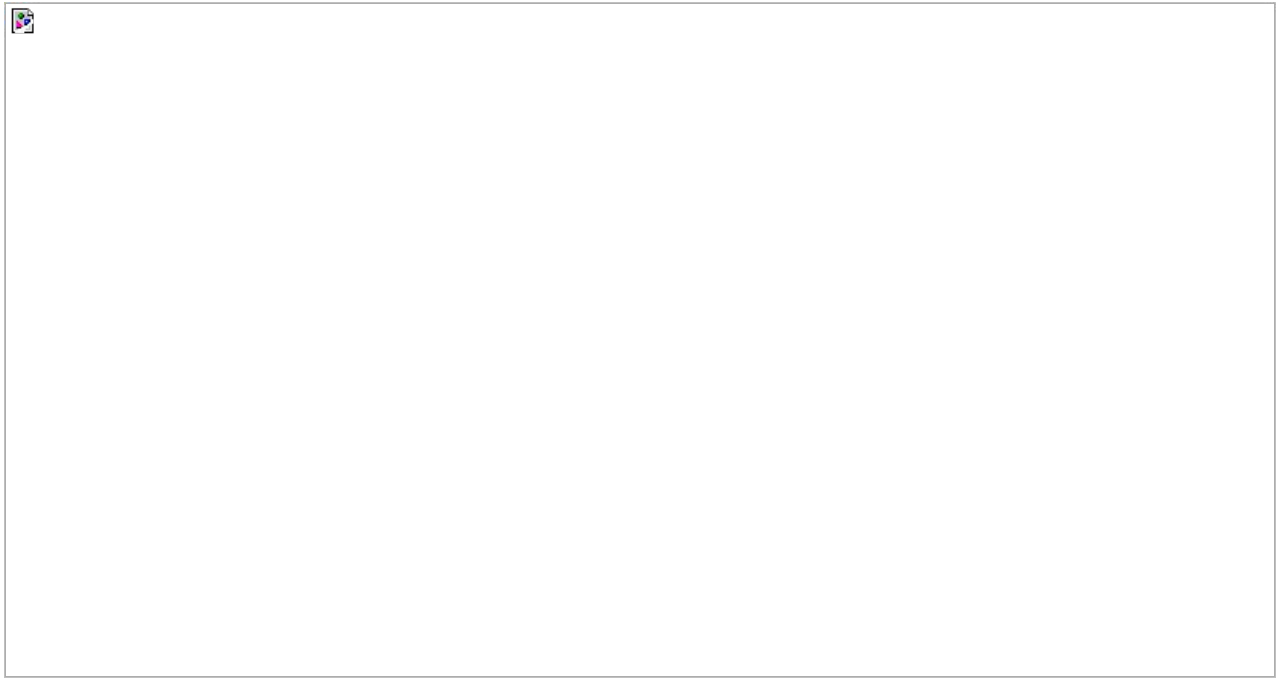


The double pushout approach



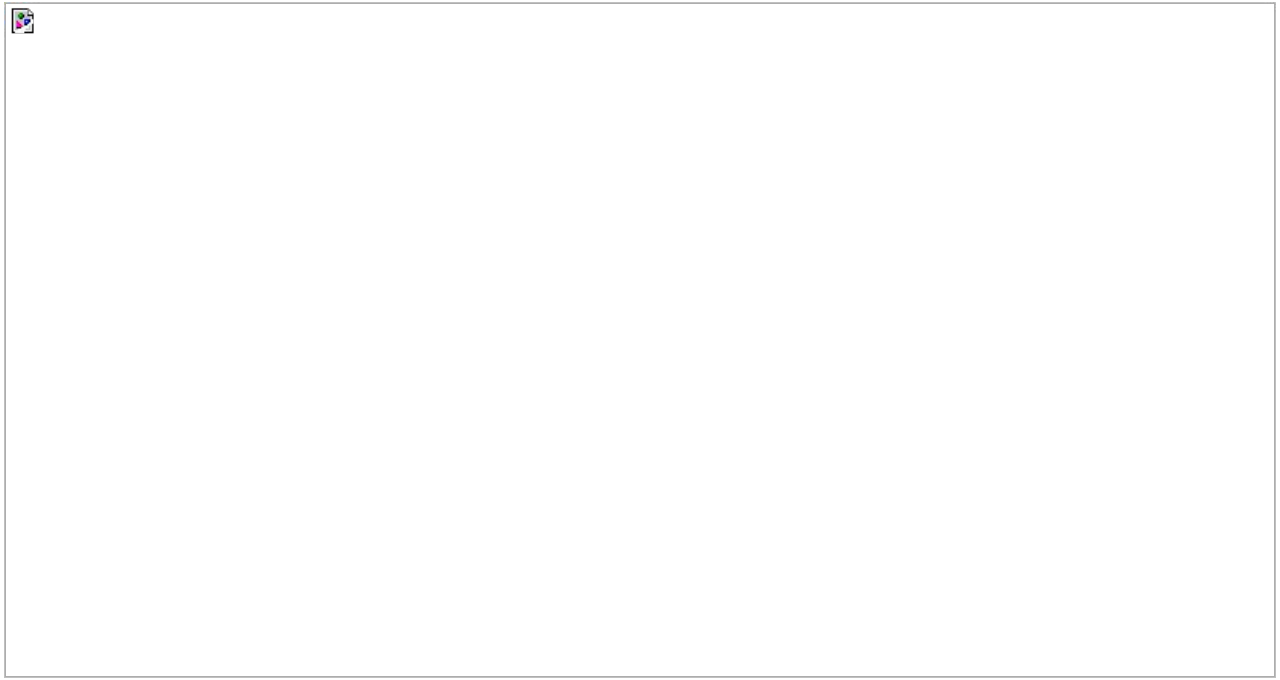
The double pushout approach

$p_{move}: L^{l} \leftarrow K^{r} \rightarrow R$



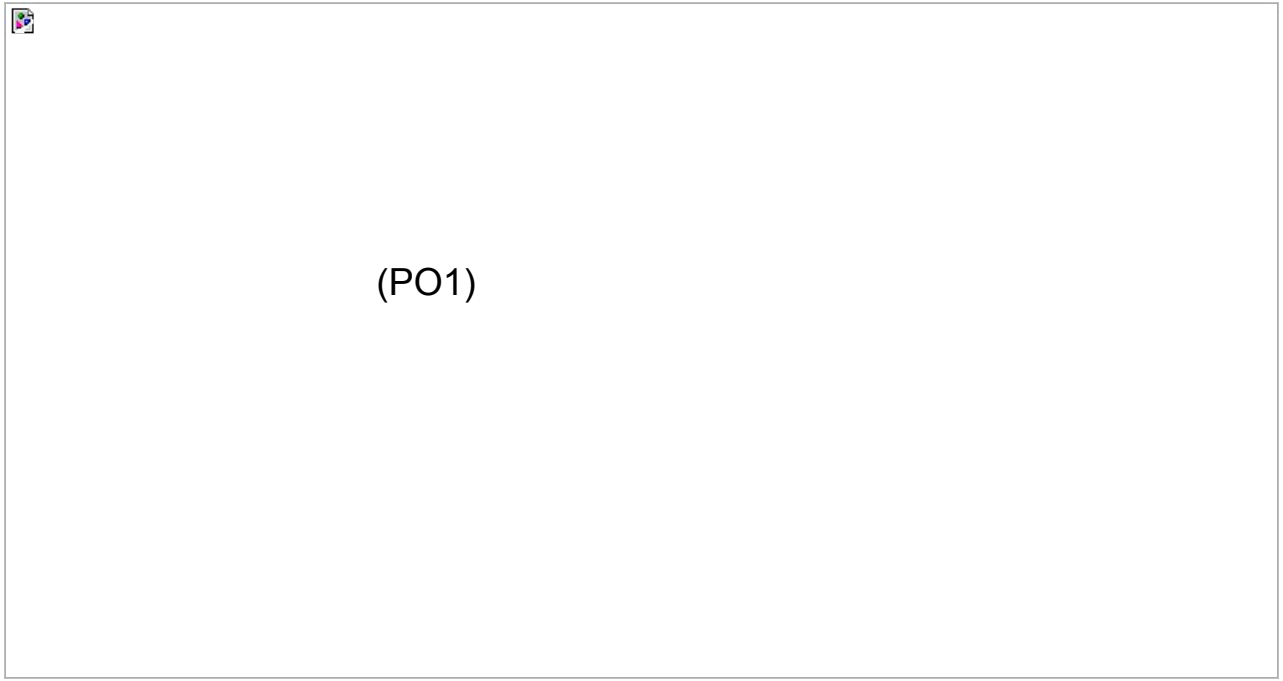
The double pushout approach

$p_{move}: L \xleftarrow{l} K \xrightarrow{r} R$



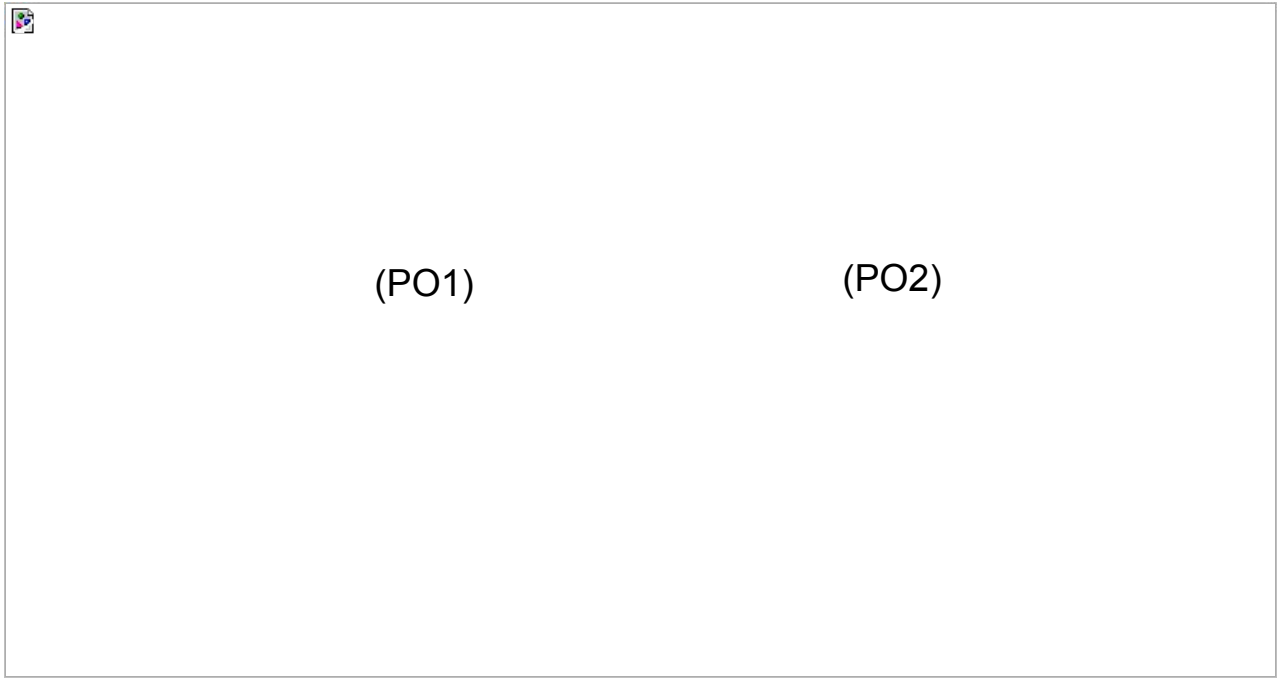
The double pushout approach

$p_{move}: L \xleftarrow{l} K \xrightarrow{r} R$



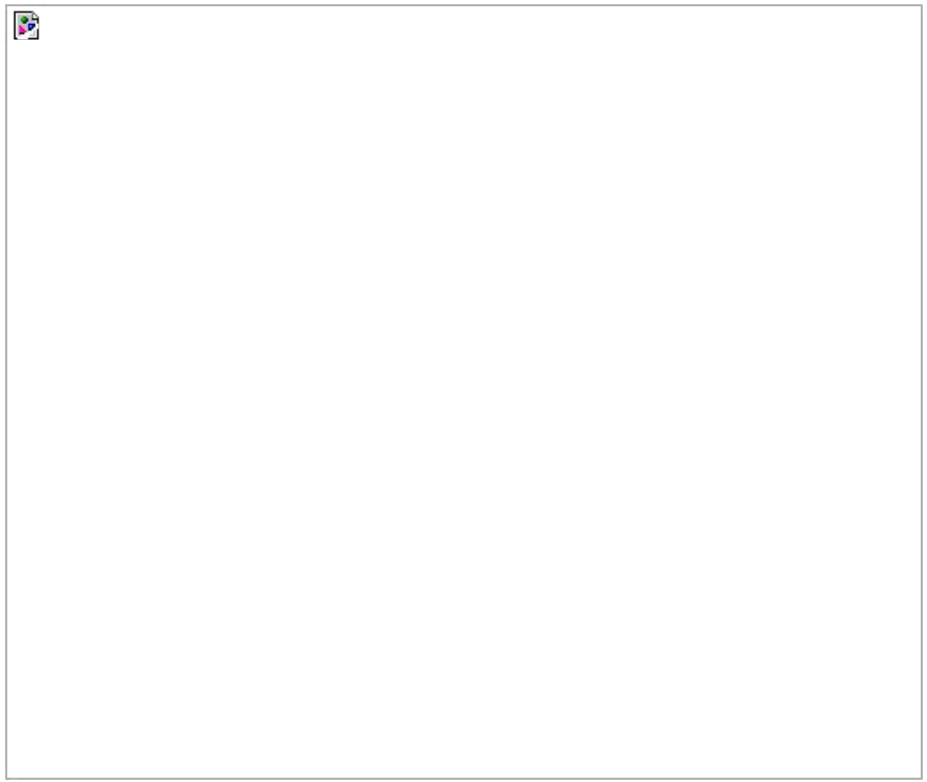
The double pushout approach

$p_{move}: L^{l} \leftarrow K^{r} \rightarrow R$



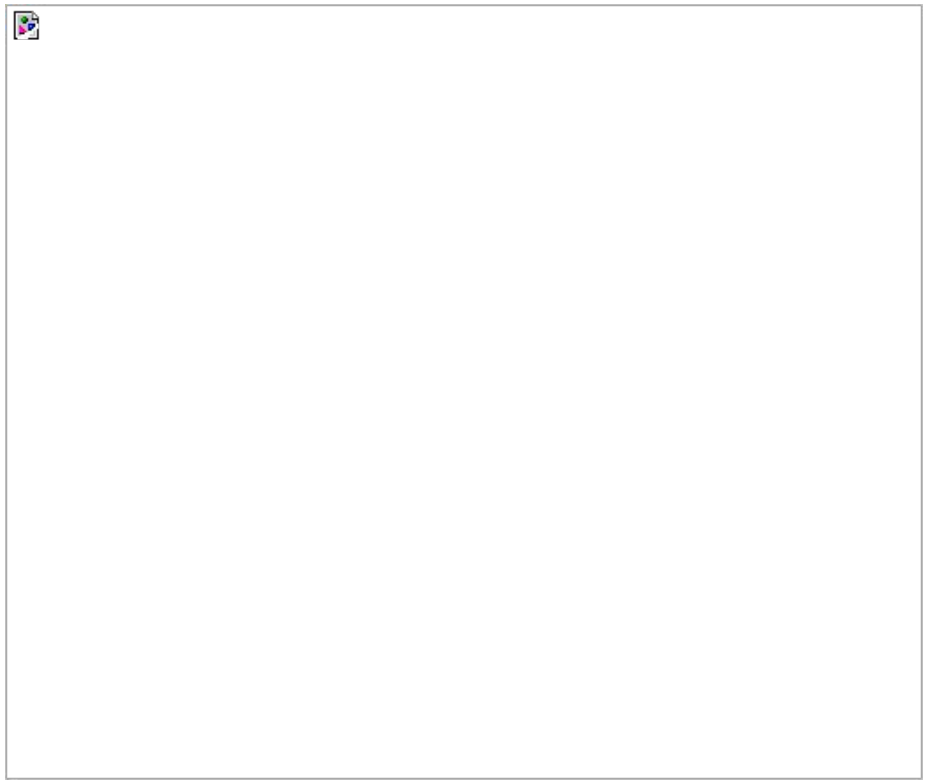
The single pushout approach

$p_{move}: L^r \rightarrow R$

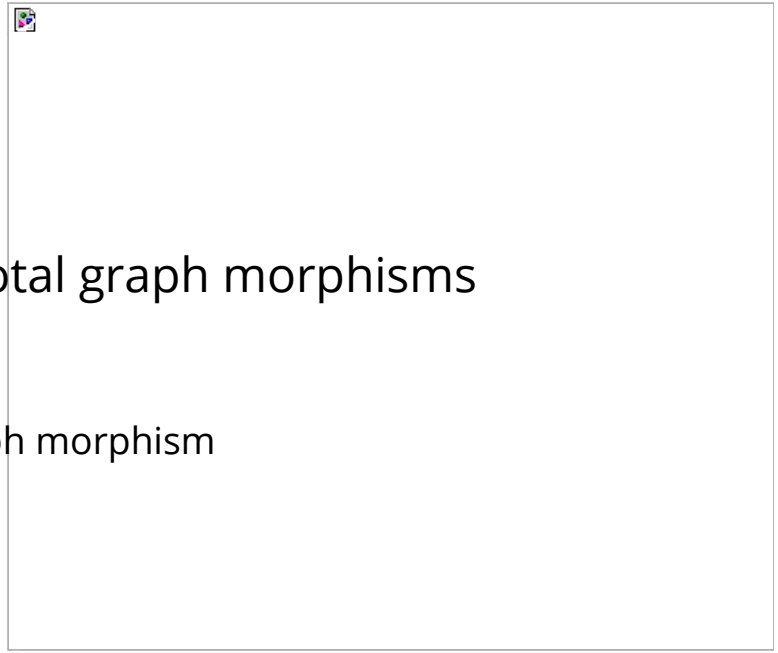


The single pushout approach

$p_{move}: L^r \rightarrow R$



The single pushout approach



1. Use **partial** graph morphisms instead of total graph morphisms
2. Introduce category `Graph_p`
   a. with partial graph morphism instead of total graph morphism
3. Define production $p_{\texttt{<name>}}$: $L \xrightarrow{r} R$
4. Construct specific co-equalizer in `Graph_p`
5. Construct pushout of two partial graph morphisms

The single pushout approach

Implemented

1. Use **partial** graph morphisms instead of total graph morphisms
2. Introduce category `Graph_p`
   a. with partial graph morphism instead of total graph morphism
3. Define productions $p_{\texttt{<name>}}$: $L \xrightarrow{r} R$
4. Construct specific co-equalizer in `Graph_p`
5. Construct pushout of two partial graph morphisms

The single pushout approach

Implemented

1. Use **partial** graph morphisms instead of total graph morphisms
2. Introduce category `Graph_p`
   a. with partial graph morphism instead of total graph morphism
3. Define productions $p_{\langle name \rangle}: L \xrightarrow{r} R$

Partially implemented

4. Construct specific co-equalizer in `Graph_p`
5. Construct pushout of two partial graph morphisms

Not yet implemented

Implementation

- `PartialFunction`
  - inherited by `TotalFunction`
- `PFinSets`
  - id and composition of `PartialFunctions`
- `PGraphMorphism`
  - inherited by `GraphMorphism`
  - `PartialFunctions` instead of `TotalFunctions`
  - reimplemented validity check
- `TGraphMorphism`
  - modified validity check
    - distinction between `PGraphMorpism` and `GraphMorphism`

Retrospective

- Implemented the basic constructs

- TODO:
  - implement co-equalizer
  - implement single pushout construction

- Rule application somewhat more intuitive than for Double Pushout
  - however: SPO construction more **complex**
  - construction of co-equalizer is **not trivial**

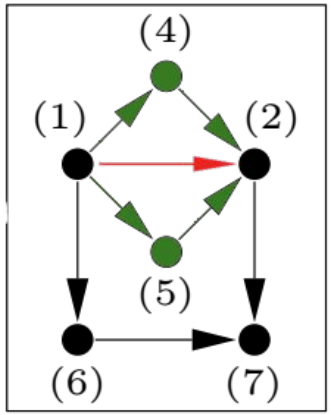
Thank you for your attention

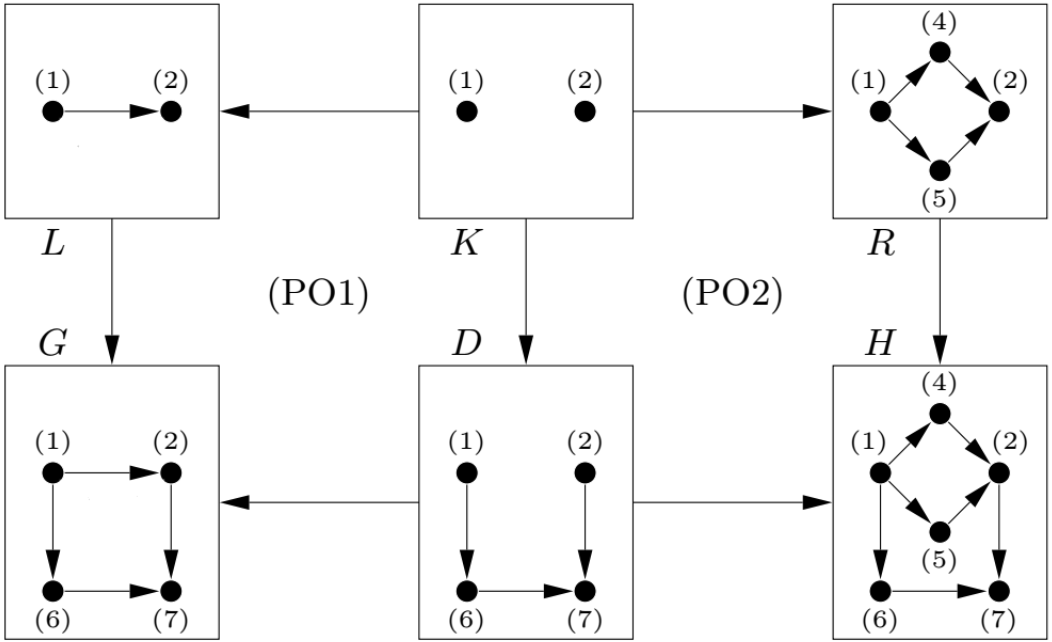
References:

Ehrig, Hartmut, et al. "Algebraic approaches to graph transformation: part ii: single pushout approach and comparison with double pushout approach." *Handbook of Graph Grammars*. 1997.

Ehrig, Ehrig, and Prange, Taentzer. "Fundamentals of algebraic graph transformation. With 41 Figures (Monographs in Theoretical Computer Science. An EATCS Series)." 2006.

The double pushout approach





The single pushout approach