AGREE – Algebraic Graph REwriting with Controlled Embedding

A. Corradini[1] and D. Duval[2] and R. Echahed[3] and F. Prost[3] and L. Ribeiro[4]

Dipartimento di Informatica, Università di Pisa,

LJK - Université de Grenoble

LIG - Université de Grenoble

INF - Universidade Federal do Rio Grande do Sul

27th of November 2014

Introduction

- Cloning is a basic operation of programming.

  $\Longrightarrow$ Typically to produce a web site one starts to copy an existing one, then one modifies it accordingly to its will.

- Several approaches for Graph Transformations show limitations: there are no easy ways to handle connecting edges.

- Typically rule:

Introduction

- Cloning is a basic operation of programming.

  $\Longrightarrow$ Typically to produce a web site one starts to copy an existing one, then one modifies it accordingly to its will.

- Several approaches for Graph Transformations show limitations: there are no easy ways to handle connecting edges.

- Typically rule:

matched to

Introduction

- Cloning is a basic operation of programming.

  $\Longrightarrow$ Typically to produce a web site one starts to copy an existing one, then one modifies it accordingly to its will.

- Several approaches for Graph Transformations show limitations: there are no easy ways to handle connecting edges.

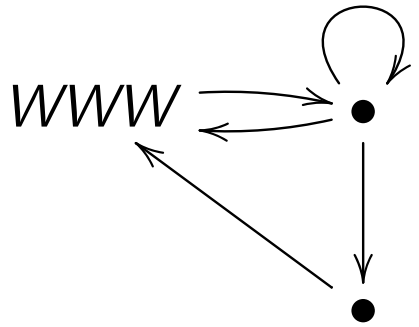
- Typically rule:

matched to

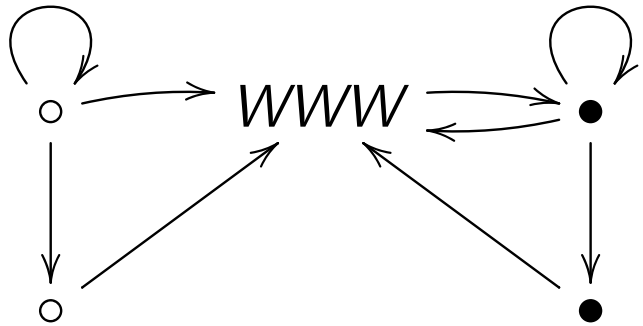
produces

Motivating example : copy of web pages

- The structure of a web site typically as two kind of links :
  - Internal links: file hierarchy (indirect link)
  - External links: references pointing outside of the site.
- The cloning of a web site consists in duplicating all local files and keeping external links shared between the two copies.



should be cloned as follows



Introduction

- In order to have a flexible to cloning:
  1. One has to be able to make the distinction between matched edges and non matched edges incident to the copied pattern.
  2. One has to be able to express what to do with incident edges not matched.

- AGREE solves these issues by :
  - Using the partial map classifier of the left-hand side.
  - Building a PB using a graph controling the embeding.

Plan

Partial map classifier

AGREE rewriting

Related works

Plan

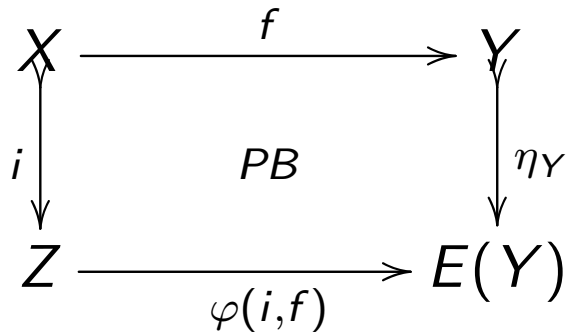
Partial map classifier

AGREE rewriting

Related works

Partial maps

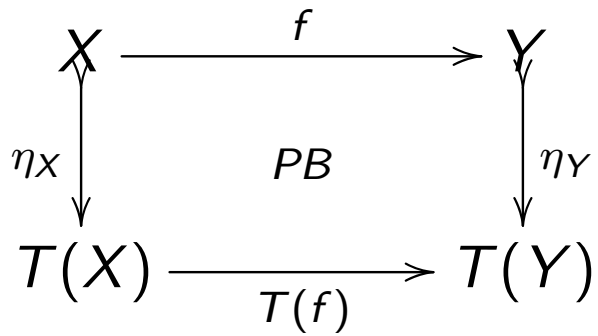
- A *partial map over* $\mathbf{C}$, denoted $(i, f) : Z \rightharpoonup Y$, is a span made of a mono $i : X \rightarrowtail Z$ and a map $f : X \to Y$.
- Composition of partial maps is defined using pullbacks in $\mathbf{C}$.
- The category of *partial maps over* $\mathbf{C}$ denoted $\mathbf{P}$.
- Let $I : \mathbf{C} \to \mathbf{P}$ be the inclusion functor wich maps $f : X \to Y$ to $(id_X, f) : X \rightharpoonup Y$
- $E : \mathbf{P} \to \mathbf{C}$ is right adjoint to $I : \mathbf{C} \to \mathbf{P}$ if there is $\eta : Id_{\mathbf{C}} \dot{\to} E \circ I$ such that:



Partial map classifier

Definition

If I has a right adjoint E, we write (T, η, μ) the monad associated with the adjunction $I \dashv E$. It is called *the partial map classifier* of $\mathbf{C}$.

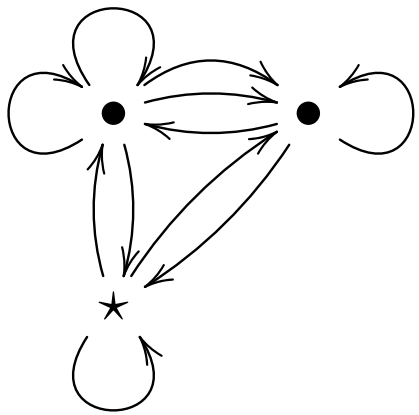
- $T = E \circ I$
- 

- Intuitively : $T(X)$ is the object allowing you to make total any partial function (i.e. you have all you need to complete any partial function).

Partial map classifier for graphs

- G is defined by E, N and $s, t : E \to N$.
- $T(G)$ is defined by $E_{T(G)}, N \uplus \{\star\}$ such that $\star_{n,p} : n \to p$ is in $E_{T(G)}$ for each pair of vertices (n, p) in $G + \{\star\}$.

$G =$ 

$T(G) =$ 

Plan

Partial map classifier

AGREE rewriting

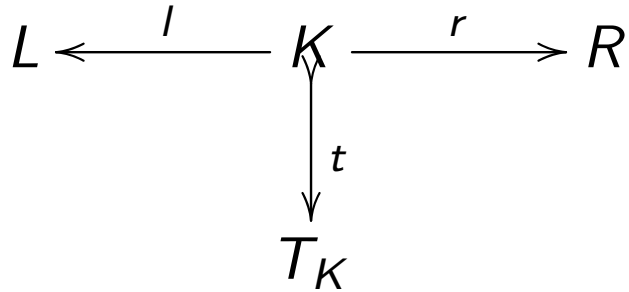
Related works

AGREE rule

Definition (AGREE rules and matches)

Let $\mathbf{C}$ with a partial map classifier (T, η, μ).

- A *rule* is



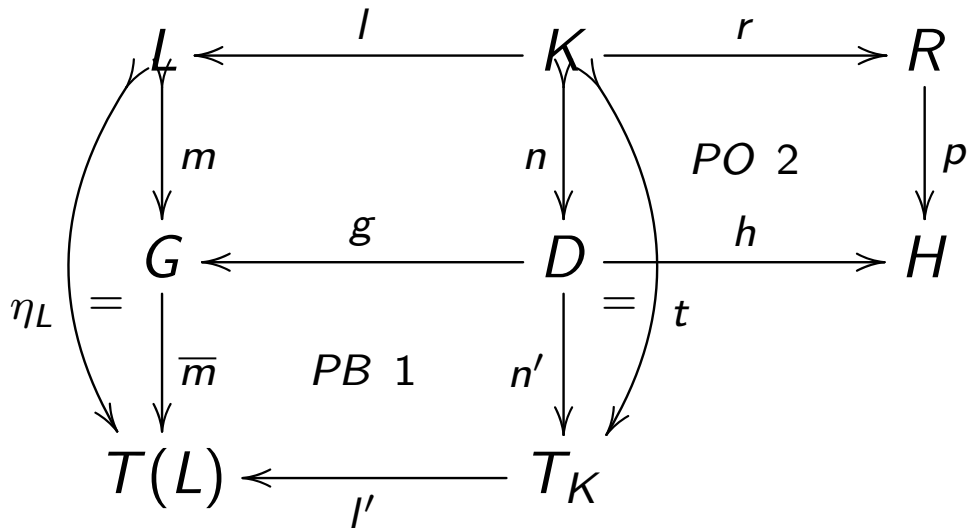
- A *match* of a rule ρ with left-hand-side $K \xrightarrow{l} L$ is a mono $L \overset{m}{\rightarrowtail} G$.

AGREE rewrite step

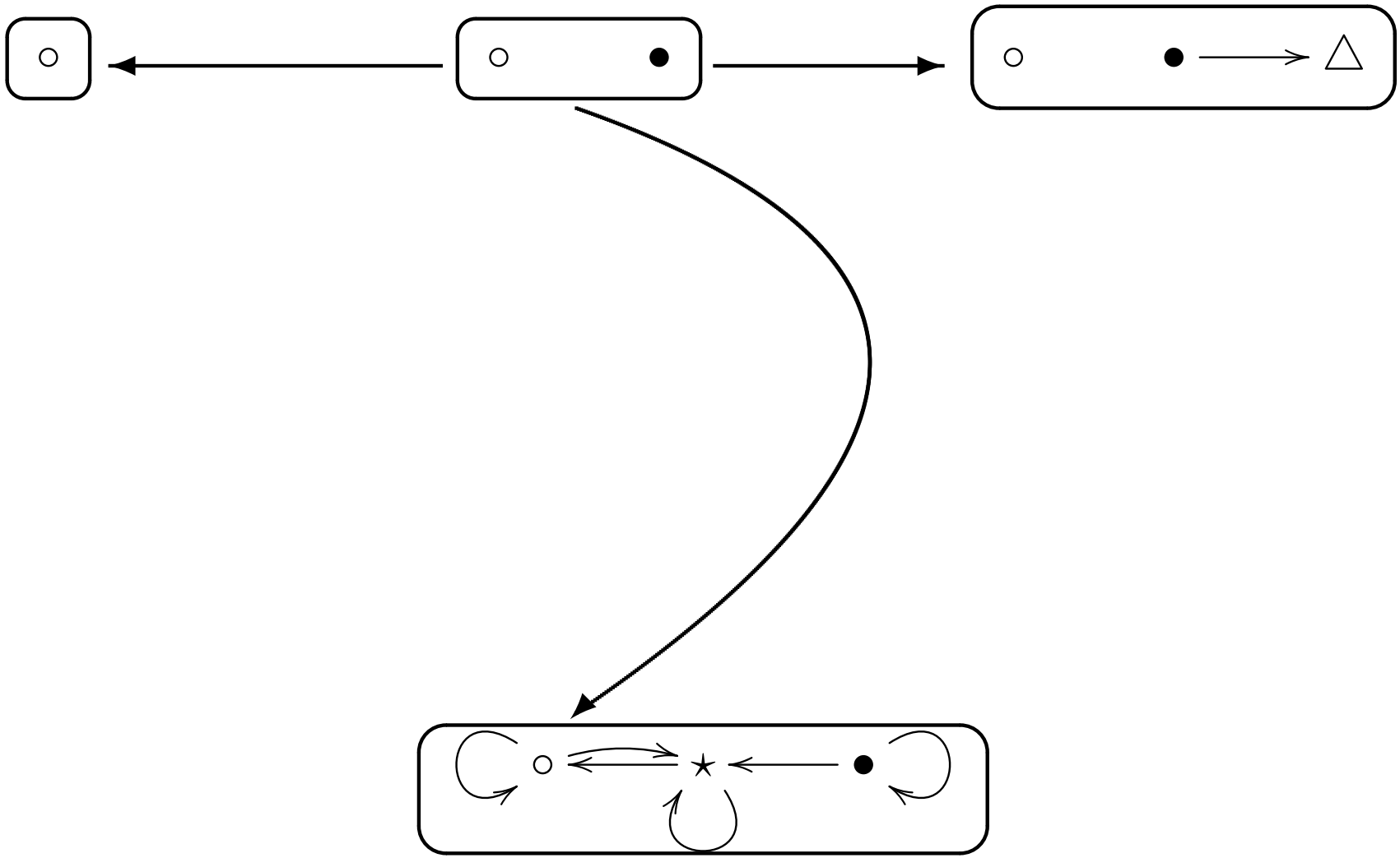
Definition (AGREE rewriting)

Given $\rho = (K \xrightarrow{l} L, K \xrightarrow{r} R, K \overset{t}{\rightarrowtail} T_K)$ and $L \overset{m}{\rightarrowtail} G$: $G \Rightarrow_{\rho,m} H$ is computed as follows:

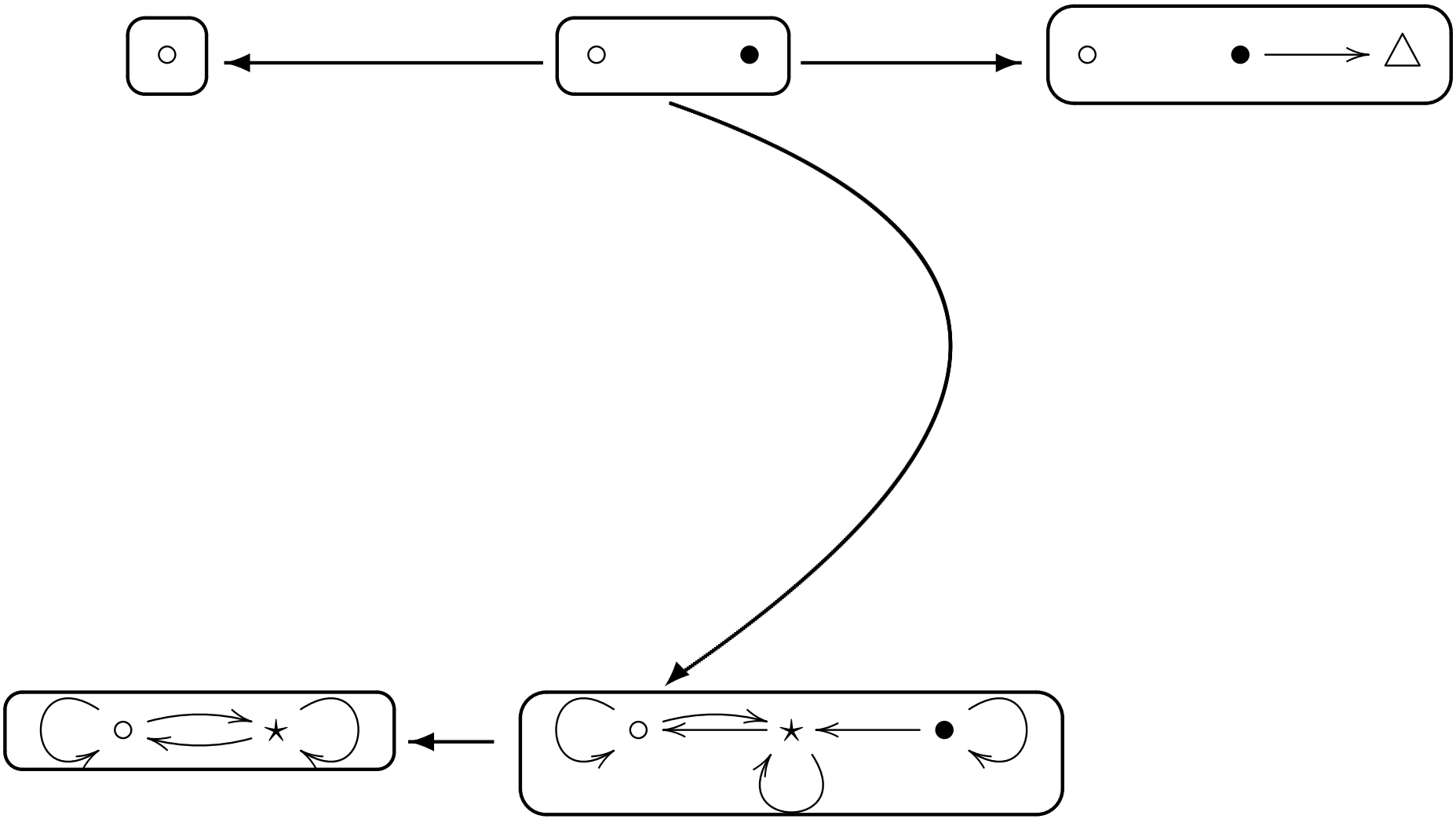
1. Span $G \xleftarrow{g} D \xrightarrow{n'} T_K$ is the pullback of $G \xrightarrow{\overline{m}} T(L) \xleftarrow{l'} T_K$. Since $l' \circ t = \eta_L \circ l$ there is a unique $K \overset{n}{\rightarrowtail} D$.
2. $R \xrightarrow{p} H \xleftarrow{h} D$ is the pushout of $D \xleftarrow{n} K \xrightarrow{r} R$.



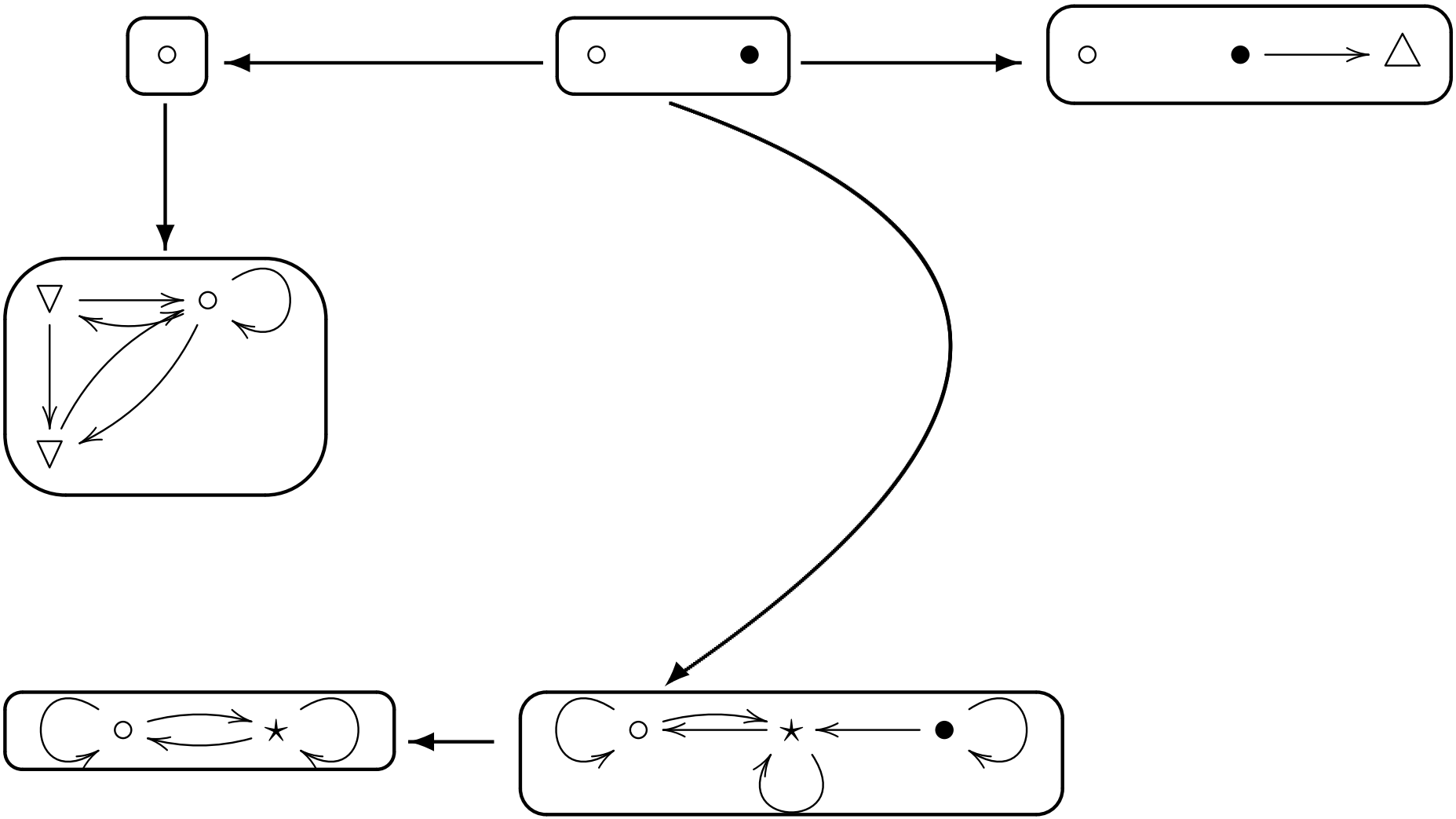
Web copy with AGREE rewriting



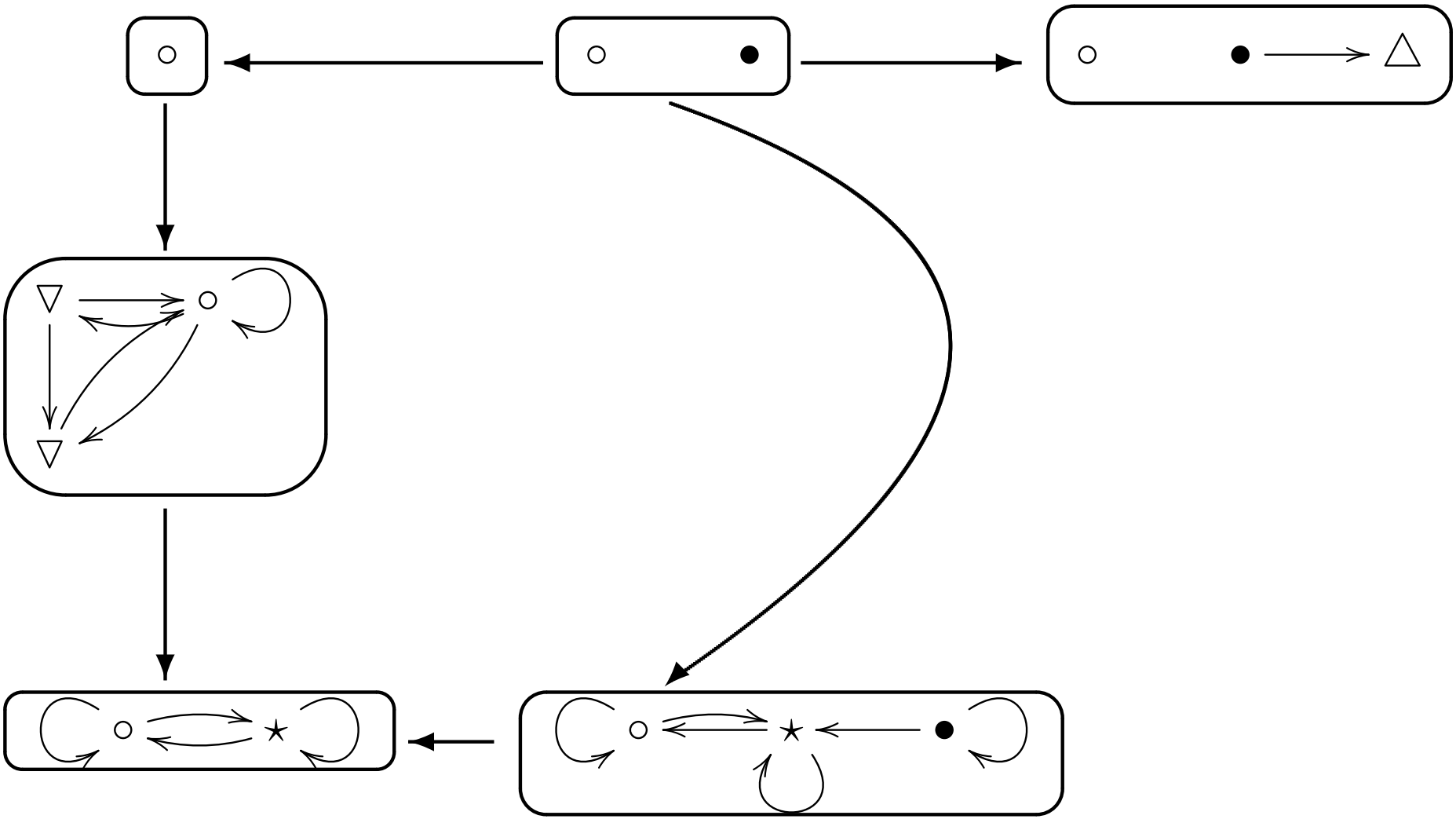
Web copy with AGREE rewriting



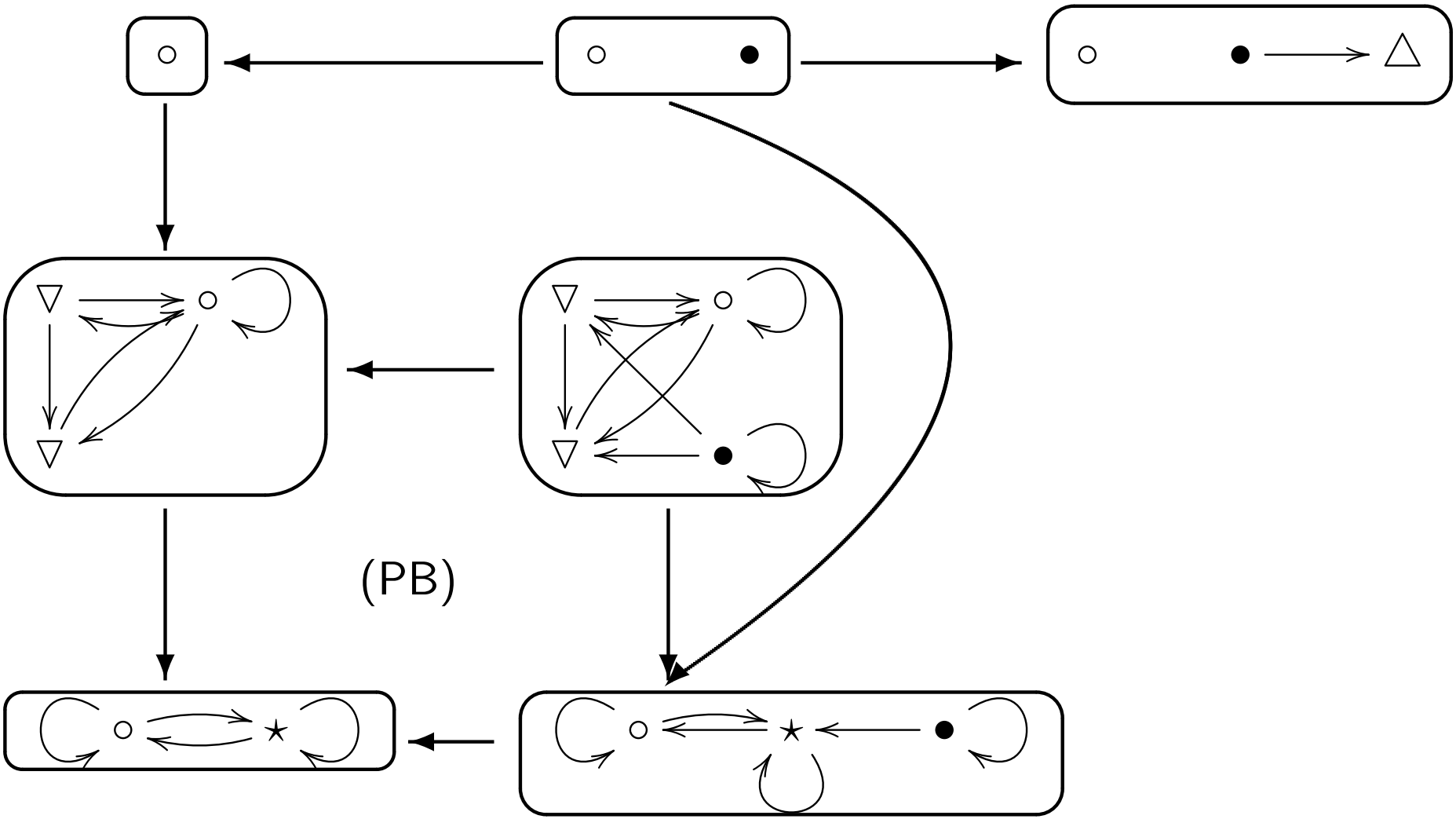
Web copy with AGREE rewriting



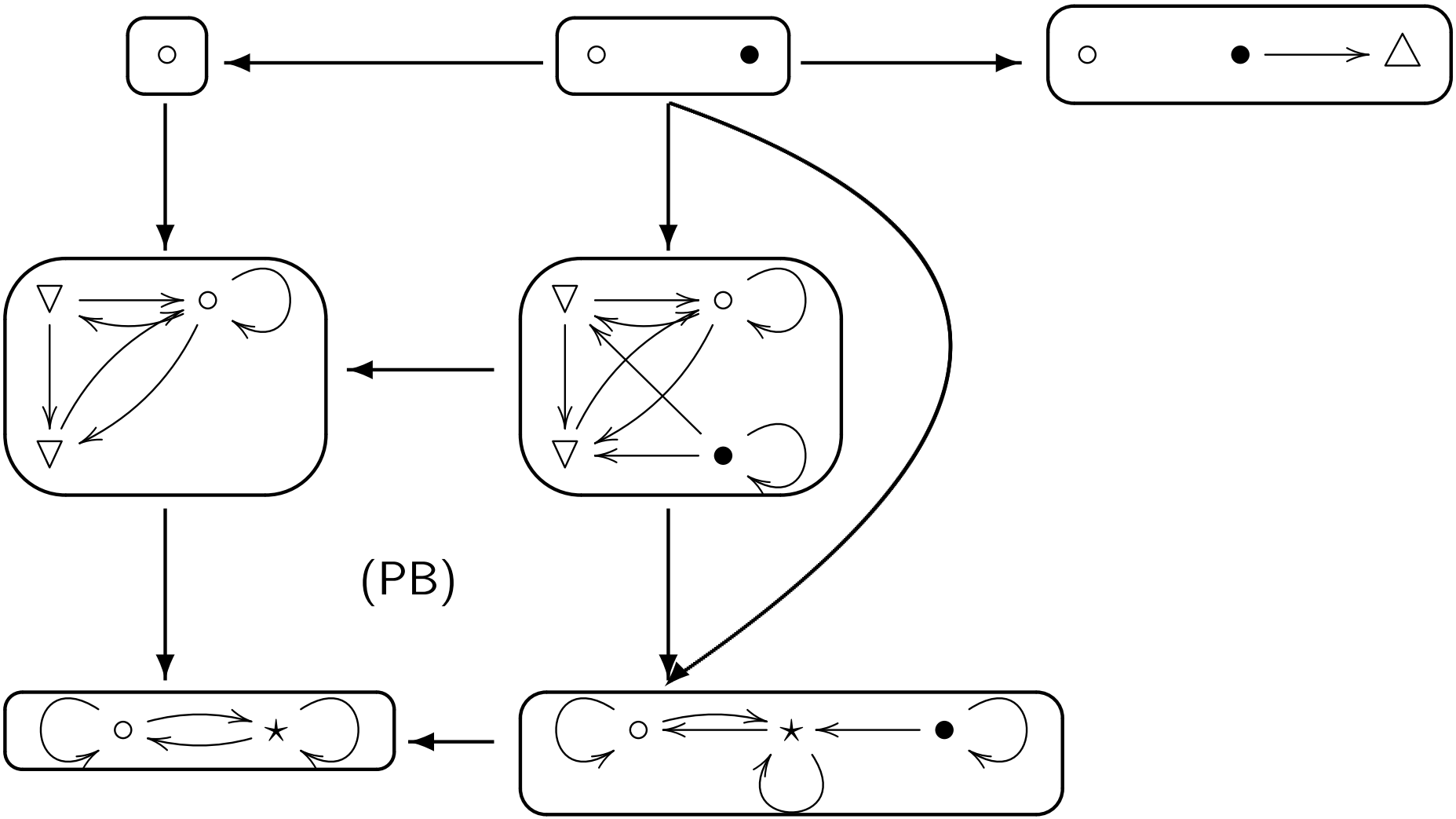
Web copy with AGREE rewriting



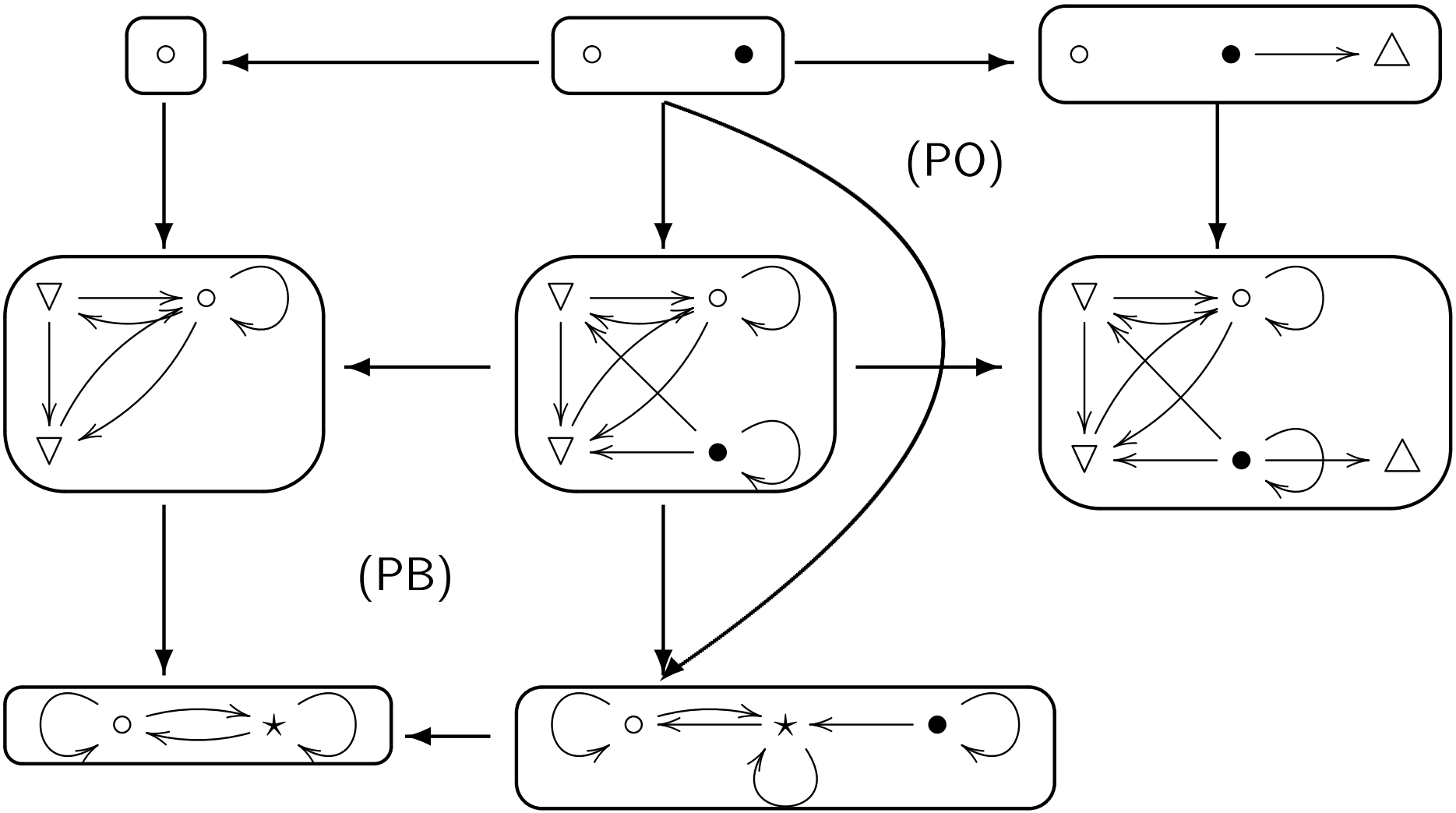
Web copy with AGREE rewriting



Web copy with AGREE rewriting



Web copy with AGREE rewriting



Locality issues

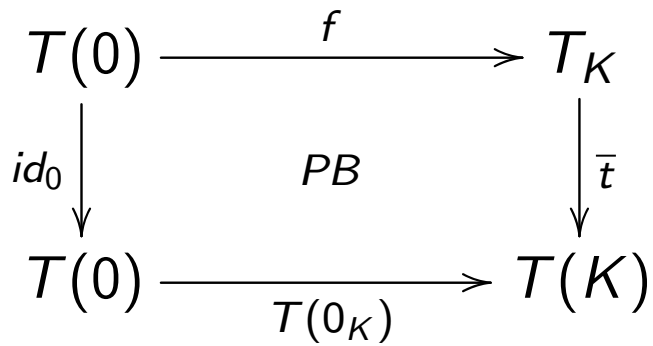
- T_K controls the way the embedding is cloned.

Locality issues

- T_K controls the way the embedding is cloned.
- If $\star$ is not in T_K everything disapears...
  two $\star$ copy the whole www...

Locality issues

- T_K controls the way the embedding is cloned.
- If $\star$ is not in T_K everything disapears...
  two $\star$ copy the whole www...
- The rule is no longer local. (curse or blessing ?).
- One solution: if there is a strict initial object 0
  and requirement for T_K:



Plan

Partial map classifier

AGREE rewriting

Related works

AGREE vs SqPO

- AGREE subsumes SqPO with injective matches.

Theorem

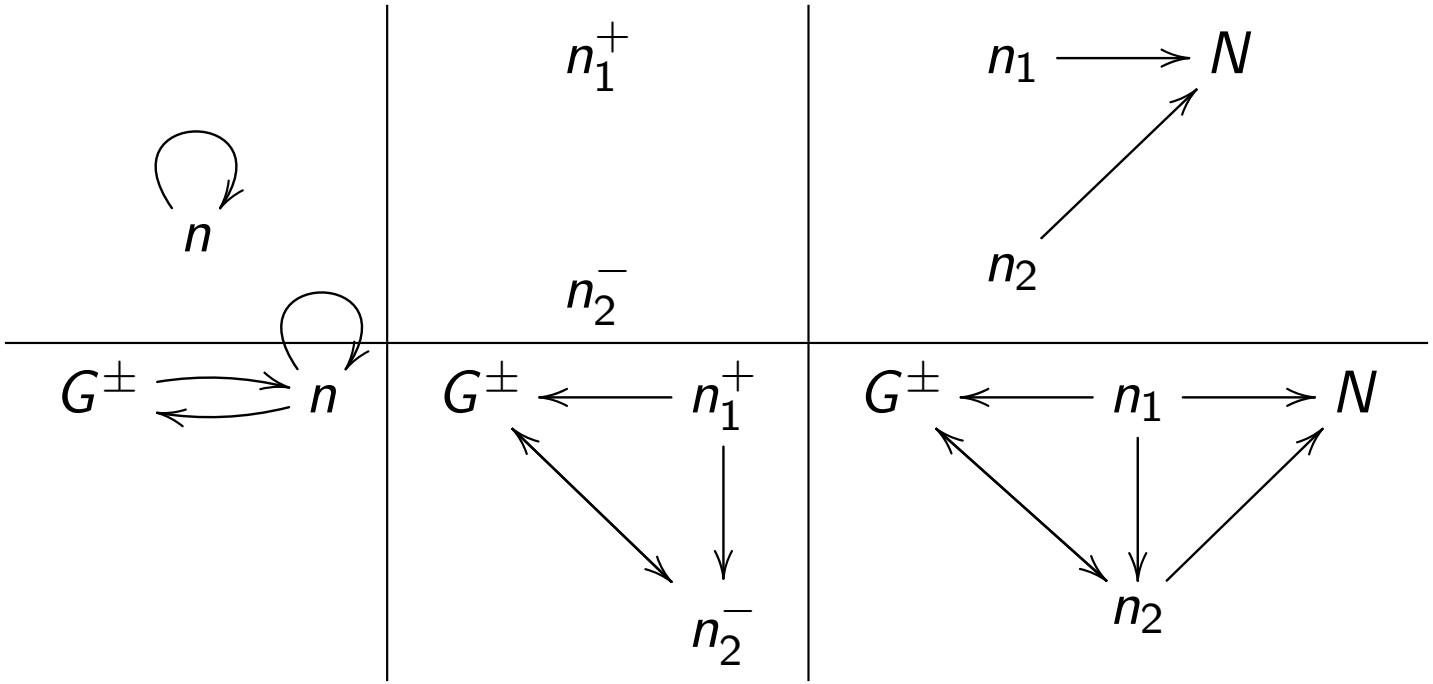
Let $\mathbf{C}$ *be a category with pullbacks and pushouts and with a partial map classifier* (T, η, μ)*. Let* $\rho = L \xleftarrow{l} K \xrightarrow{r} R$ *be a rule and* $m : L \rightarrowtail G$ *be a monic match. Then*

$$G \Rightarrow^{SqPO}_{\rho,m} H \qquad \text{if and only if} \qquad G \Rightarrow^{AGREE}_{(l,k,\eta_K),m} H$$

In words, the application of rule ρ *to match* m *using the SqPO approach has exactly the same effect of applying to* m *the same rule enriched with the embedding* $K \xrightarrow{\eta_K} T(K)$ *using the AGREE approach.*

AGREE vs PSqPO

- Polarized sesqui-pushout adds a limited control to cloning :
  - Nodes are decorated with any combination of "+" and "-".
  - Edges are only allowed from "+" to "-".
- SQPO in a polarized graph is controled by annotations in K



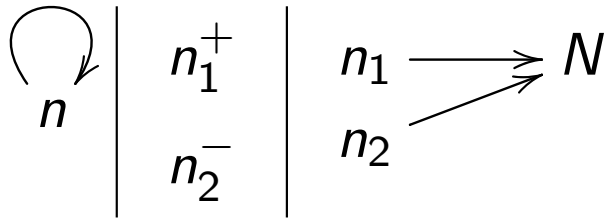
AGREE vs PSqPO

Theorem

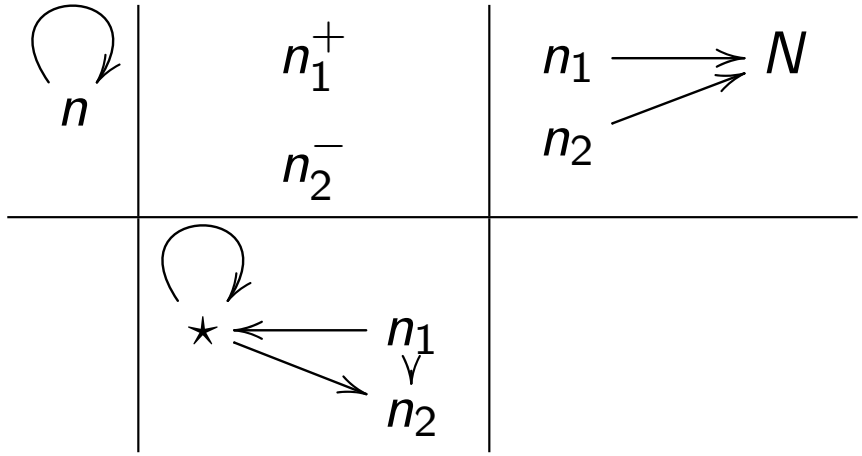
For each PSqPOrule ρ made of $L \xleftarrow{l} K \xrightarrow{r} R$ and $\mathbb{K} = (K, N_K^+, N_K^-)$, let $T_K = \mathrm{Depol}(\mathbb{T}(\mathbb{K}))$ and $t = \mathrm{Depol}(\eta_{\mathbb{K}})$. Then we get an AGREE rule $\rho^{\pm}$ in **Gr** *by adding to the span $L \xleftarrow{l} K \xrightarrow{r} R$ the embedding $t : K \rightarrowtail T_K$. For each mono $m : L \rightarrowtail G$ in* **Gr**, *applying the PSqPOrule ρ to m provides the same graph H as applying the AGREE rule $\rho^{\pm}$ to m in* **Gr**.

AGREE vs PSqPO

The PSqPO rule:



is implemented by the AGREE following rule:



Conclusion

- AGREE-rewriting is very flexible and subsumes many proposals.
- In some sense it is more moral (to me at least) than previous proposals since it is the combination of dual aspects :

PB	PO
clone	merge
delete	add
comatch	match
global	local

- Many formalization possible for a rule and a step (to study).
- Non-local applications usefull ?