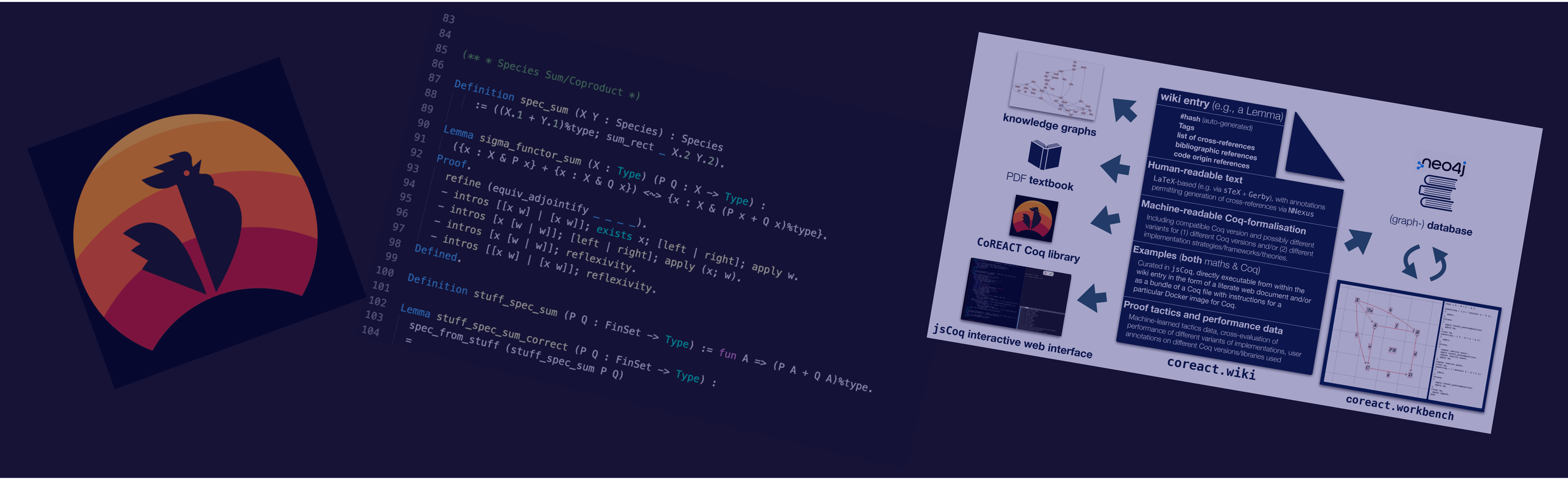
CoREACT

Coq-based Rewriting: towards Executable Applied Category Theory

Consortium: IRIF (UP), LIP (ENS-Lyon), LIX (École Polytechnique), Sophia-Antipolis (Inria)



INSTITUT DE RECHERCHE EN INFORMATIQUE FONDAMENTALE

cnrs

Université Paris Cité

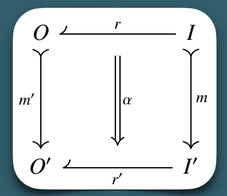
Coq WG session on User Interfaces

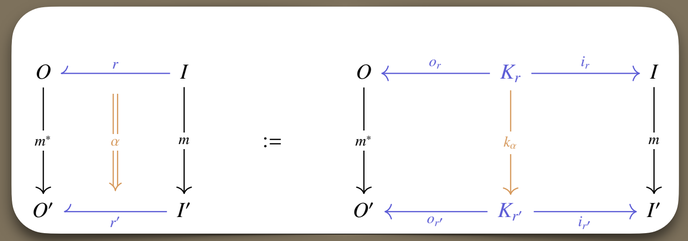
January 31, 2023

Nicolas Behr

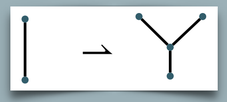
CNRS, Université Paris Cité, IRIF (UMR 8243)

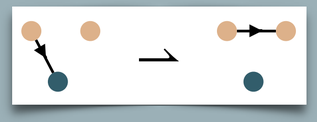
compositional rewriting double categories (crDCs)



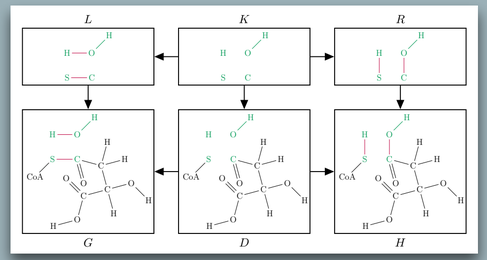


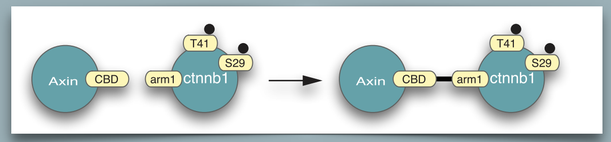
Explicit **rewriting semantics** (DPO, SqPO, …)





organic chemistry





biochemistry

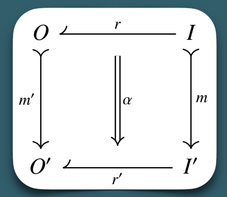
Instantiations of rewriting semantics in theory and applications

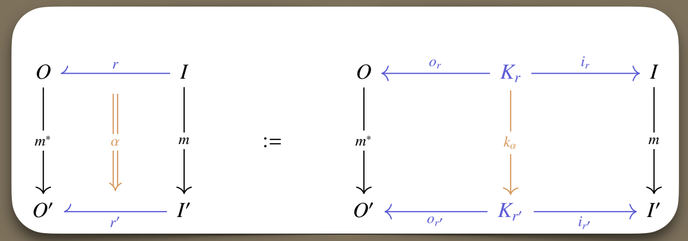
Fundamentals of Compositional Rewriting Theory*

Nicolas Behr[a,*], Russell Harmer[b], Jean Krivine[a]

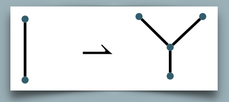
[a] *Université Paris Cité, CNRS, IRIF, 8 Place Aurélie Nemours, Paris Cedex 13, 75205, France*
[b] *Université de Lyon, ENS de Lyon, UCBL, CNRS, LIP, 46 allée d'Italie, Lyon Cedex 07, 69364, France*

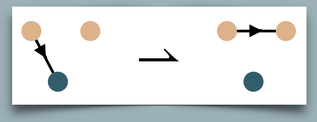
compositional rewriting double categories (crDCs)



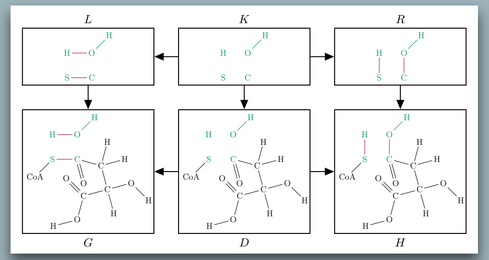


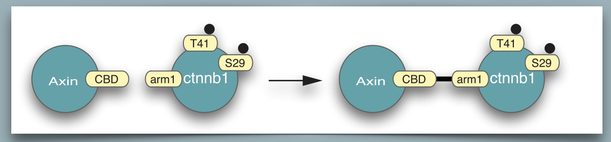
Explicit **rewriting semantics** (DPO, SqPO, …)





organic chemistry





biochemistry

Instantiations of rewriting semantics in theory and applications

?

Fundamentals of Compositional Rewriting Theory*

Nicolas Behr[a,*], Russell Harmer[b], Jean Krivine[a]

[a] *Université Paris Cité, CNRS, IRIF, 8 Place Aurélie Nemours, Paris Cedex 13, 75205, France*
[b] *Université de Lyon, ENS de Lyon, UCBL, CNRS, LIP, 46 allée d'Italie, Lyon Cedex 07, 69364, France*

nLab

Home Page All Pages

species

Contents

https://ncatlab.org

CoREACT

Coq-based Rewriting: towards Executable Applied Category Theory

Consortium: IRIF (UP), LIP (ENS-Lyon), LIX (École Polytechnique), Sophia-Antipolis (Inria)

Partner	Last name	First name	Position
Université Paris Cité	BEHR	Nicolas	CNRS CR
	GALLEGO	Emilio	Inria SRP
	GHEERBRANT	Amélie	MdC
	HERBELIN	Hugo	Inria DR
	MELLIÈS	Paul-André	CNRS DR
	ROGOVA	Alexandra	PhD st.
	PhD student	(to recruit)	
ENS-Lyon	HARMER	Russell	CNRS CR
	HIRSCHOWITZ	Tom	CNRS DR
	POUS	Damien	CNRS DR
	PostDoc	(to recruit)	
École Polytechnique	MIMRAM	Samuel	PR
	WERNER	Benjamin	PR
	ZEILBERGER	Noam	MdC
	PostDoc	(to recruit)	
Inria Sophia-Antipolis	BERTOT	Yves	Inria DR
	COHEN	Cyril	Inria CR
	TASSI	Enrico	Inria CR
	PostDoc	(to recruit)	
Cambridge University	LAFONT	Ambroise	PostDoc

Project type / duration / start date	**PRC / 48 months / March 2023**
Budget	**€424335**
Scientific coordinator	**Nicolas BEHR**

coreact.wiki

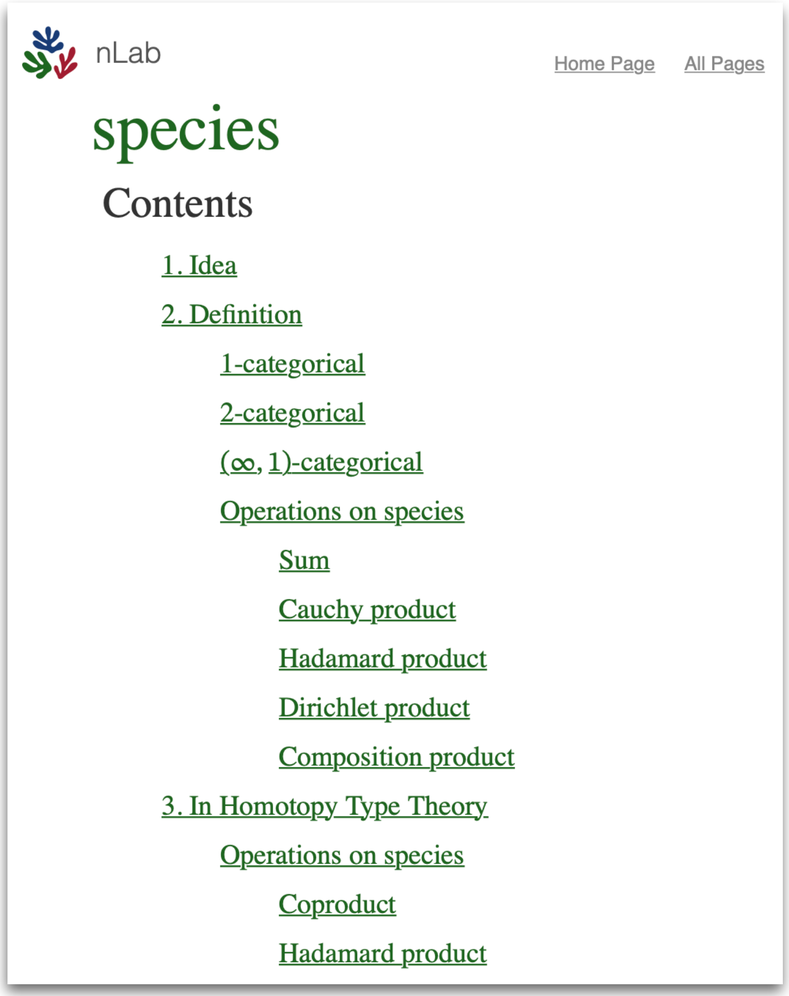
Main objectives of the CoREACT/GReTA ExACT initiative

- Development of a methodology for **diagrammatic reasoning in Coq**
- **Formalization** (in Coq) and **certification** of a representative collection of axioms and theorems for **compositional categorical rewriting theory**
- Development of a **Coq-enabled interactive database and wiki system**
- Development of a CoREACT wiki-based **“proof-by-pointing” engine**
- Executable **reference prototype algorithms** from categorical structures in Coq (via the use of SMT solvers/theorem provers such as Z3)

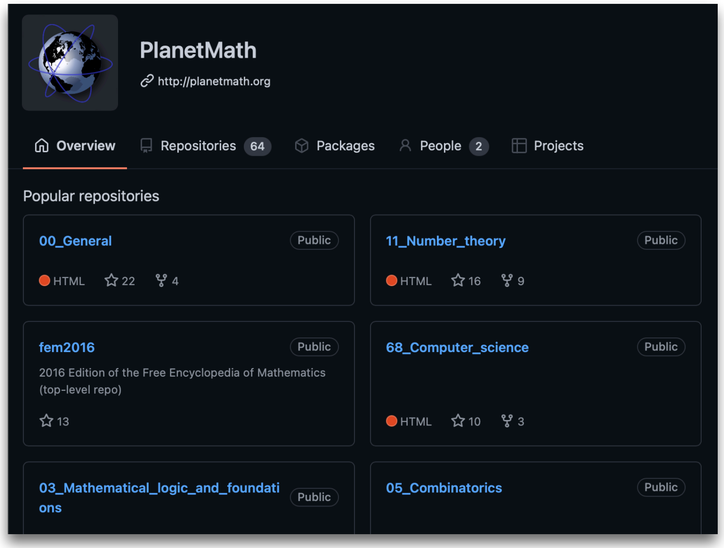
Main objectives of the CoREACT/GReTA ExACT initiative

- Development of a methodology for **diagrammatic reasoning in Coq**
- **Formalization** (in Coq) and **certification** of a representative collection of axioms and theorems for **compositional categorical rewriting theory**
- Development of a **Coq-enabled interactive database and wiki system**
- Development of a CoREACT wiki-based **“proof-by-pointing” engine**
- Executable **reference prototype algorithms** from categorical structures in Coq (via the use of SMT solvers/theorem provers such as Z3)

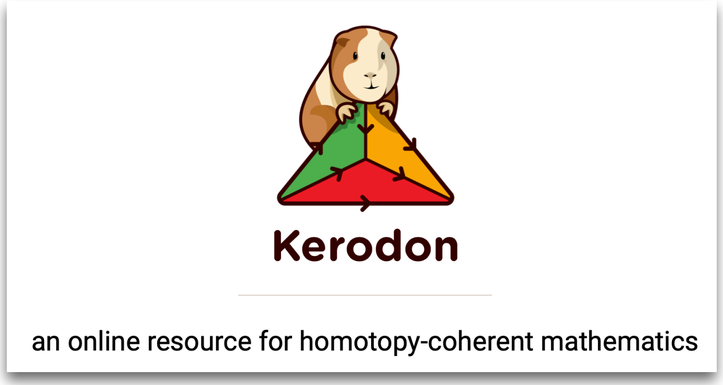
A (very non-exhaustive!) view on **wiki systems in mathematics/ (A)CT**



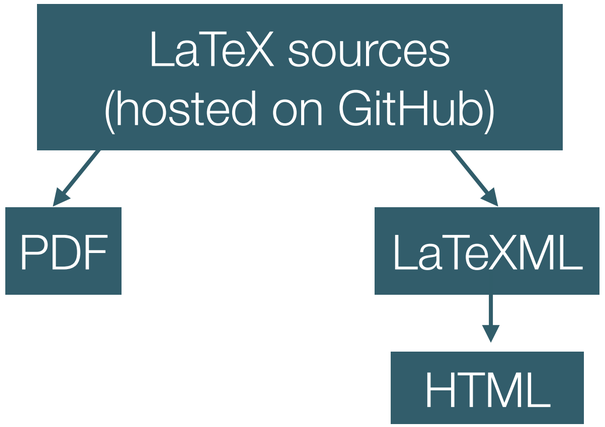
https://ncatlab.org



https://planetmath.org

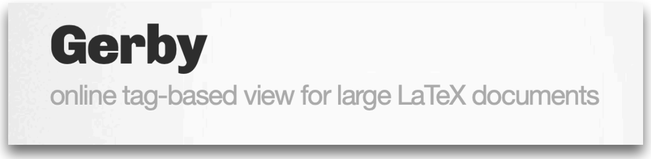


https://kerodon.net



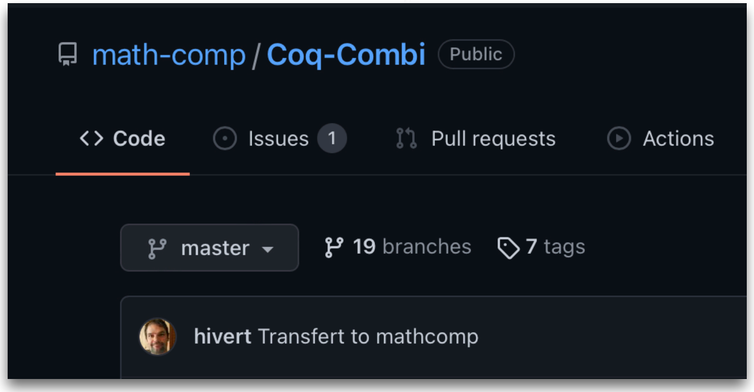
(semi-) automatic cross-linking
provided via `NNexus` system

- J. Lurie's **online textbook** on categorical homotopy theory
- technology based upon **online tags view** via the `Gerby` system

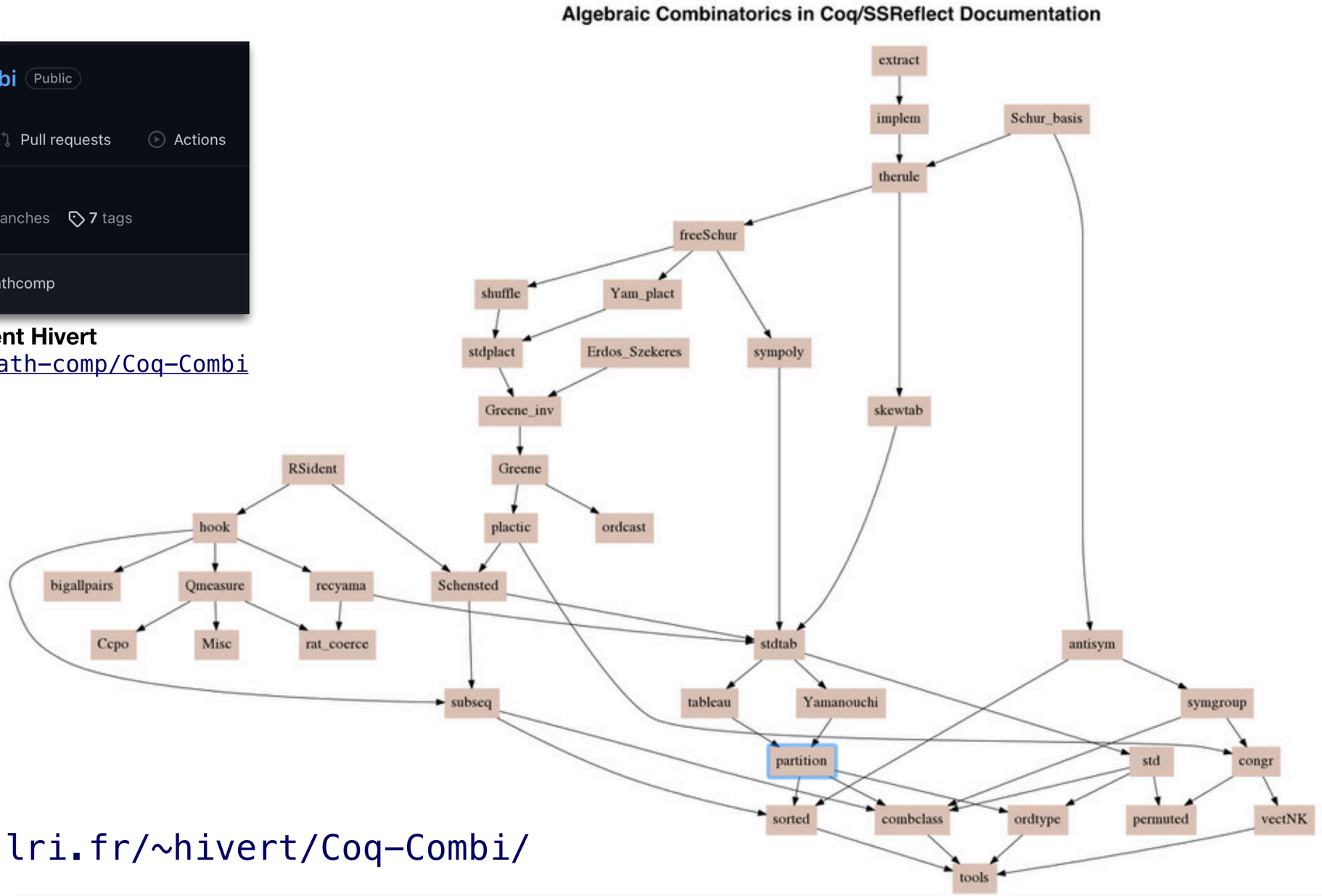


https://gerby-project.github.io

Coq-combi

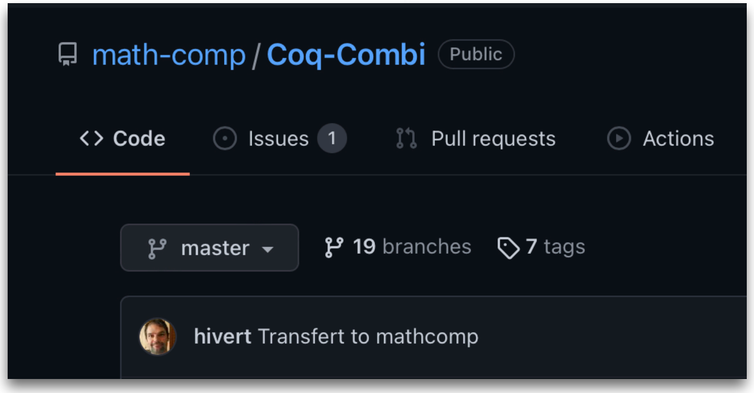


Author: **Florent Hivert**
https://github.com/math-comp/Coq-Combi



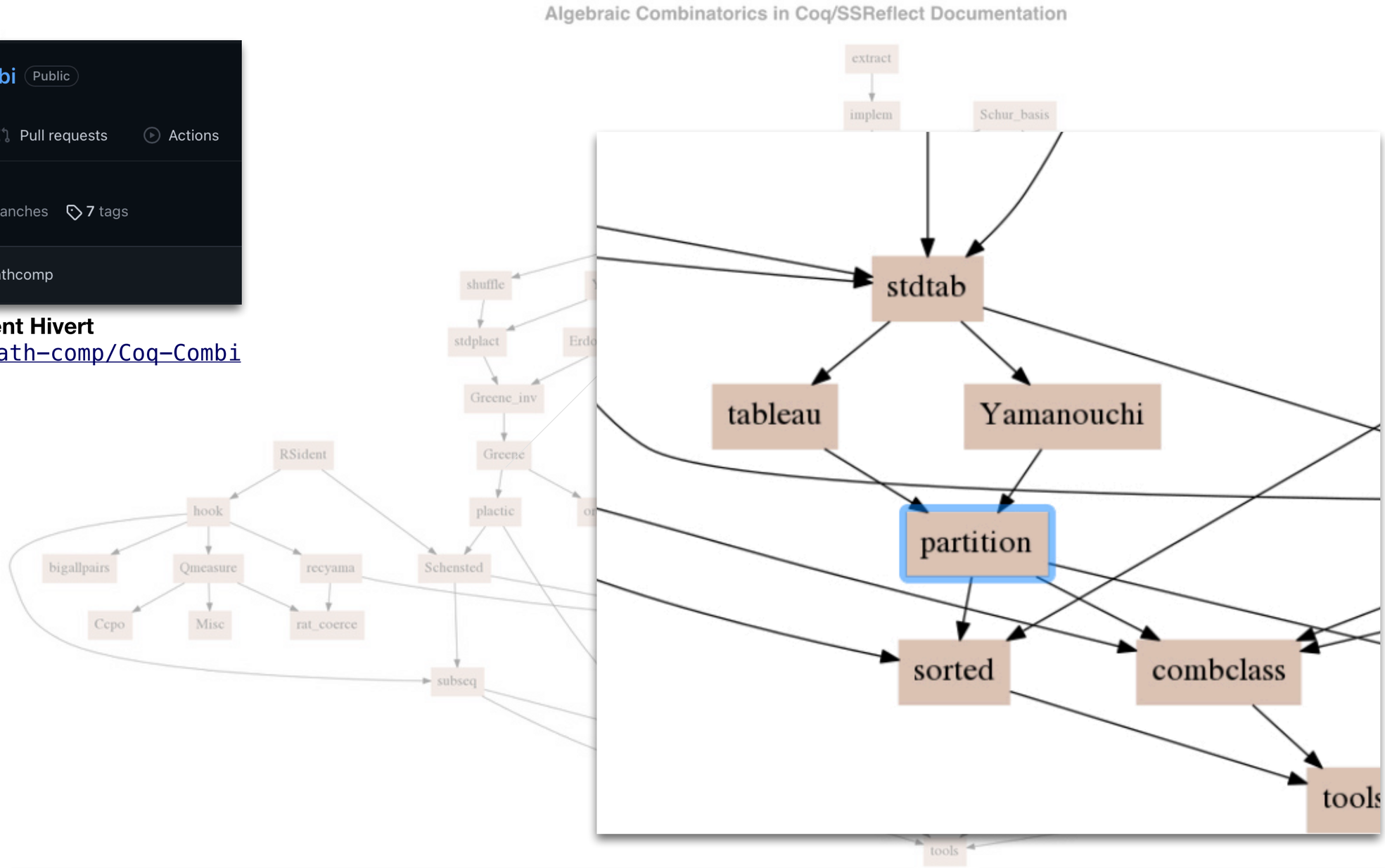
https://www.lri.fr/~hivert/Coq-Combi/

Coq-combi



Author: **Florent Hivert**

https://github.com/math-comp/Coq-Combi



Coq-combi

Combi.Combi.partition: Integer Partitions

- Shapes and Integer Partitions
- Shapes
  - A finite type `finType` for coordinate of boxes inside a shape
  - Rewriting bigops running along the boxes of a shape
  - Adding a box to a shape
- Integer Partitions
  - Definitions and basic properties
  - Corners, adding and removing corners
  - Conjugate of a partition
  - Partial sum of partitions
- Inclusion of Partitions and Skew Partitions
- Sigma Types for Partitions
- Counting functions
- TODO: Generalize and move in finOrdType

Coq-combi

Shapes and Integer Partitions

Partitions (and more generally shapes) are stored by terms of type `seq (seq nat)`. We define the following predicates and operations on `seq (seq nat)`: `(r, c)` is in `sh` if `r < sh[i]`

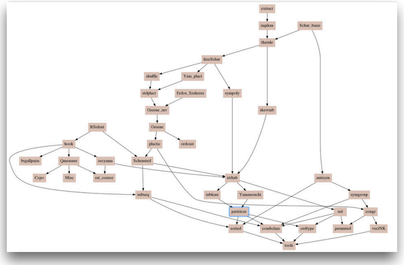
- `is_in_shape sh r c` == the box with coordinate (r, c) belongs to the shape sh, that is: `c < sh[r]`.
- `is_box_in_shape (r, c)` == uncurried version: same as `is_in_shape sh r c`.
- `box_in sh` == a sigma type for boxes in sh : `{ b | is_box_in_shape sh b }` is is canonically a `subFinType`.
- `enum_box_in sh` == a full duplicate free list of the boxes in sh.

Integer Partitions:

- `is_part sh` == `sh` is a partition
- `rem_trail0 sh` == remove the trailing zeroes of a shape
- `is_add_corner sh i` == i is the row of an addable corner of sh
- `is_rem_corner sh i` == i is the row of a removable corner of sh
- `incr_nth sh i` == the shape obtained by adding a box at the end of the i-th row. This gives a partition if i is an addable corner of sh (Lemma `is_part_incr_nth`)
- `decr_nth sh i` == the shape obtained by removing a box at the end of the i-th row. This gives a partition if i is an removable corner of sh

Sigma Types for Partitions

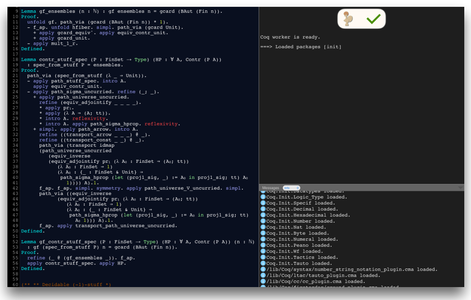
```
Section PartCombClass.

Structure intpart : Type := IntPart {pval :> seq nat; _ : is_part pval}.
Canonical intpart_subType := Eval hnf in [subType for pval].
Definition intpart_eqMixin := Eval hnf in [eqMixin of intpart by <:].
Canonical intpart_eqType := Eval hnf in EqType intpart intpart_eqMixin.
Definition intpart_choiceMixin := Eval hnf in [choiceMixin of intpart by <:].
Canonical intpart_choiceType := Eval hnf in ChoiceType intpart intpart_choiceMixin.
Definition intpart_countMixin := Eval hnf in [countMixin of intpart by <:].
Canonical intpart_countType := Eval hnf in CountType intpart intpart_countMixin.

Lemma intpartP (p : intpart) : is_part p.

Hint Resolve intpartP.

Canonical conj_intpart p := IntPart (is_part_conj (intpartP p)).

Lemma conj_intpartK : involutive conj_intpart.

Lemma intpart_sum_inj (s t : intpart) :
  (∀ k, part_sum s k = part_sum t k) → s = t.

Fixpoint enum_partnsk sm sz mx : (seq (seq nat)) :=
  if sz is sz.+1 then
    flatten [seq [seq i :: p | p <- enum_partnsk (sm - i) sz i] | i <- iota 1 (minn sm mx)]
  else if sm is sm.+1 then [::] else [:: [::]].
Definition enum_partns sm sz := enum_partnsk sm sz sm.
Definition enum_partn sm := flatten [seq enum_partns sm sz | sz <- iota 0 sm.+1 ].
```



knowledge graphs

PDF **textbook**

CoREACT Coq library



jsCoq interactive web interface

wiki entry (e.g., a Lemma)

#hash (auto-generated)
Tags
list of cross-references
bibliographic references
code origin references

Human-readable text

LaTeX-based (e.g. via sTeX + Gerby), with annotations permitting generation of cross-references via NNexus

Machine-readable Coq-formalisation

Including compatible Coq version and possibly different variants for (1) different Coq versions and/or (2) different implementation strategies/frameworks/theories.

Examples (**both** maths & Coq)

Curated in jsCoq, directly executable from within the wiki entry in the form of a literate web document and/or as a bundle of a Coq file with instructions for a particular Docker image for Coq.

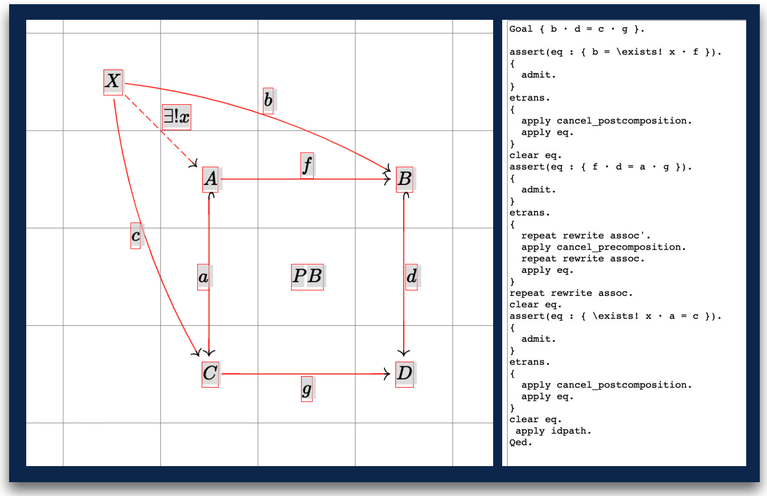
Proof tactics and performance data

Machine-learned tactics data, cross-evaluation of performance of different variants of implementations, user annotations on different Coq versions/libraries used

coreact.wiki

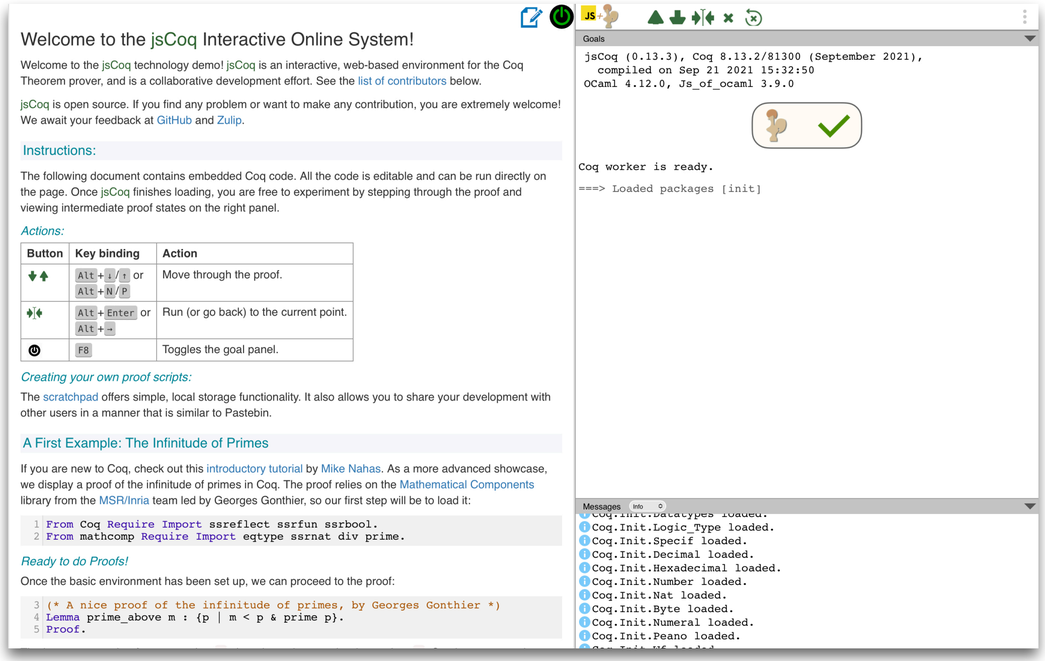
neo4j

(graph-) **database**

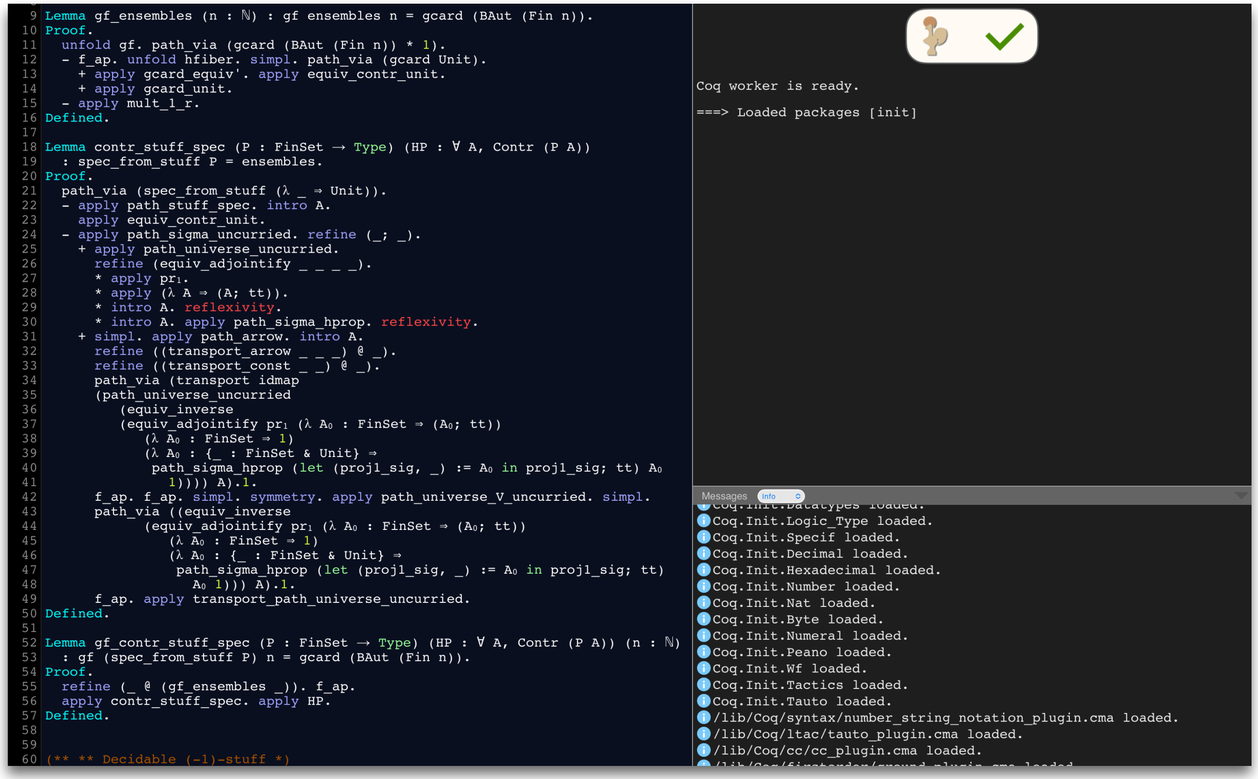


coreact.workbench

jsCoq



https://github.com/jscoq



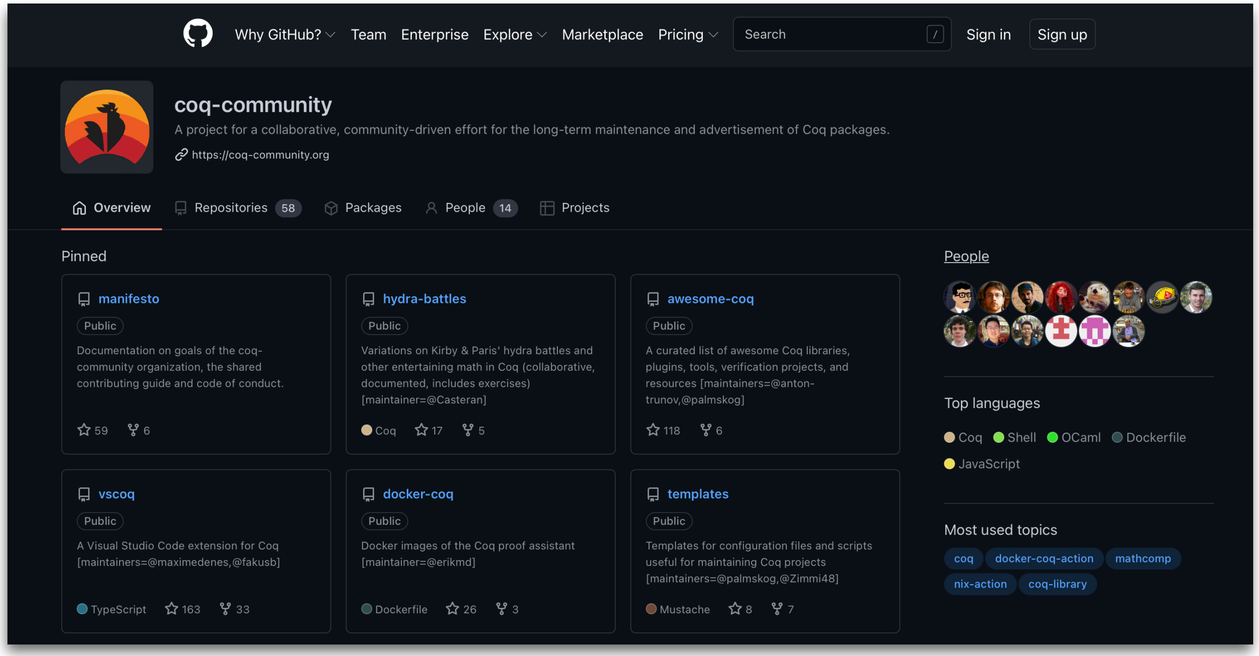
Core developer team

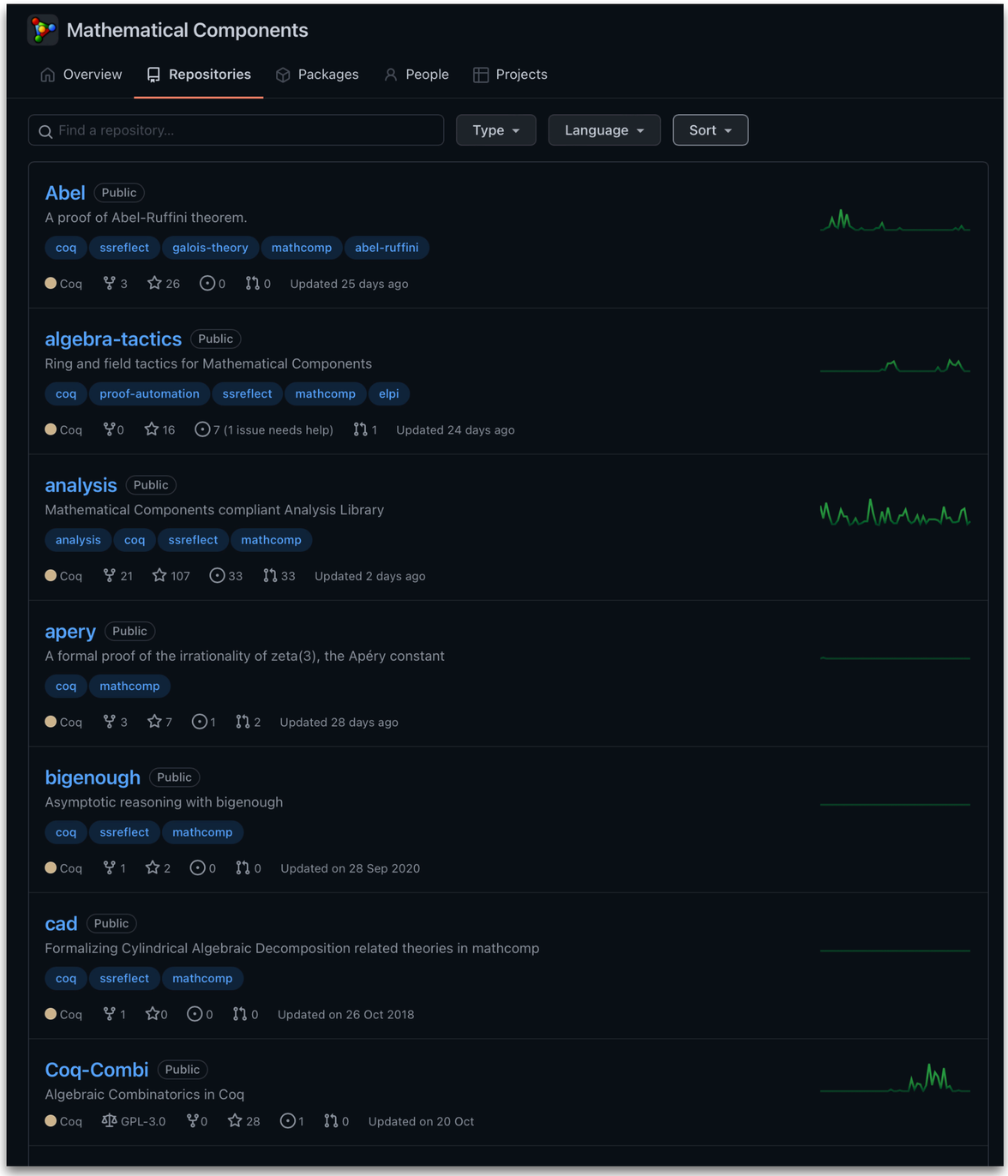
- Emilio Jesús Gallego Arias , Inria, Université de Paris, IRIF
- Shachar Itzhaky , Technion

Past Contributors

- Benoît Pin, CRI, MINES ParisTech

Coq-community





https://github.com/coq-community

*"A project for a **collaborative**, **community-driven** effort for the **long-term maintenance** and **advertisement** of Coq packages."*

- **58 repositories**

Category theory

- **Special types of categories:**
  - adhesive/quasi-adhesive/adhesive HLR/weak adhesive HLR/…
  - quasi-topoi
- **Double categories**
- **Universal constructions:**
  - stable systems of monics, factorisation systems, multi-sums, …
  - pushouts, pullbacks, final pullback complements, multi-initial pushout complements, final pullback complement augmentations, …
  - Grothendieck fibrations/multi-opfibrations/residual multi-opfibrations …
- **Lemmata on special properties of universal constructions:**
  - (De-)composition properties
  - fibrational properties
  - Beck-Chevalley conditions

Diagrammatic reasoning

- **Commutative diagrams**
- **Reasoning moves**
  - from universal properties
  - from diagrammatic lemmata
- **Compositionality of reasoning moves**

Formalisations for `coreact.workbench`

- **Auxiliary tactics** to convert between **drawings** and **Coq expressions**
- From **drawing transformations** to **reasoning moves**
- From **drawing transformations** to **Cypher queries**

Foundations of compositional rewriting theory

- **Compositional rewriting double categories (crDCs)**
- **Concurrency Theorems**
- **Associativity Theorems**
- **Rule Algebra and Stochastic Mechanics**
- **Tracelet Hopf Algebras and Decomposition Spaces**

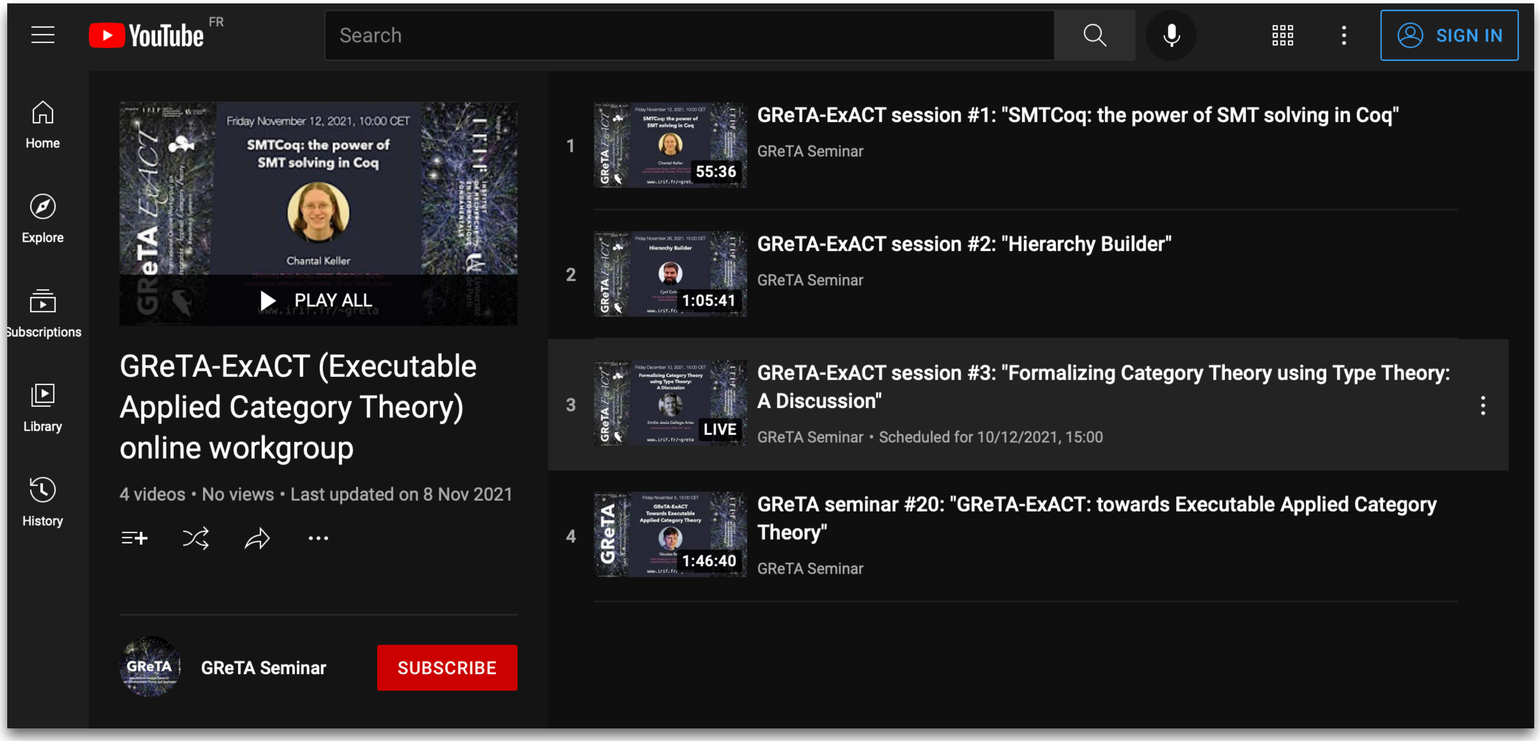
Collection of rewriting semantics

- **Double Pushout/Sesqui-Pushout/Single-Pushout/AGREE/PBPO+/…**
  - linear/input-linear/output-linear/non-linear/…
- **Theory of constraints and application conditions:**
  - nested application conditions
  - constraint-guaranteeing/-preserving semantics
- **Compositional rewriting for rules with conditions**
  - shift and transport constructions
  - Concurrency and associativity theorems
  - Rule algebras/stochastic mechanics/tracelets/…

Executable Applied Category theory (ExACT)

- **Constructive characterization of categories with adhesivity/quasi-topoi:**
  - Artin gluing/slice/coslice/product/sum/**functor and comma categories**/…
  - **collection** of practically relevant examples (**Graph** as **presheaf topos**, **SimpleGraph** via **Artin gluing**, **HyperGraph** as **comma category**, …)
- **Translation** from **rewriting semantics** to **SMT solvers/theorem provers**
- **Reference prototype algorithms for concrete rewriting semantics**

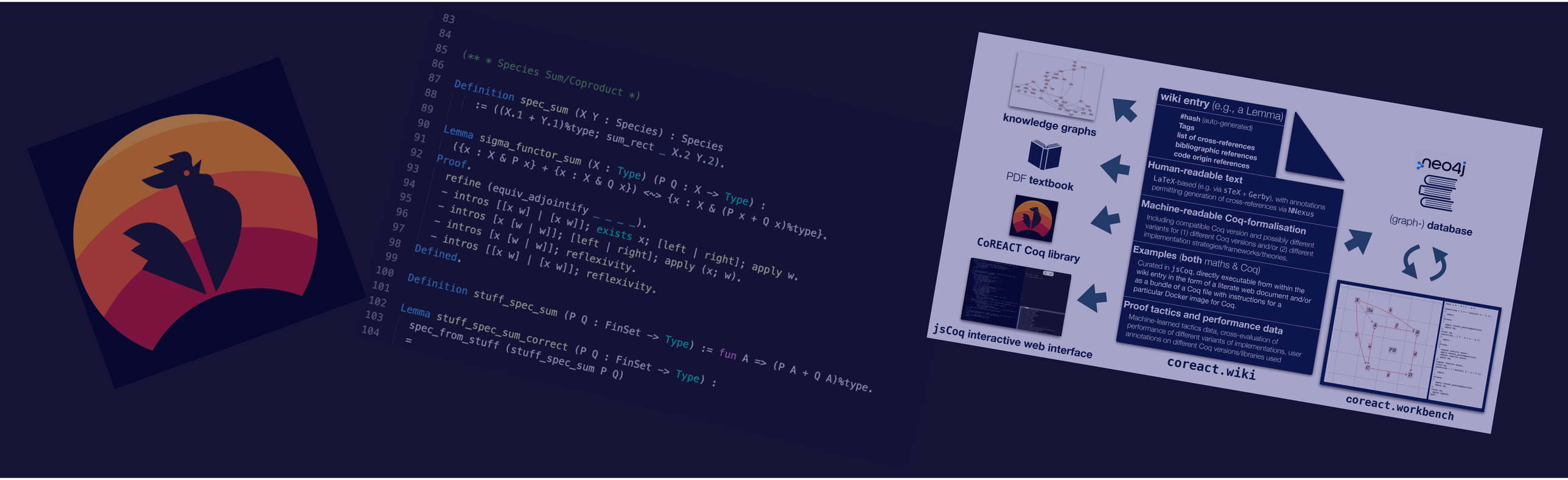
coreact.wiki



CoREACT

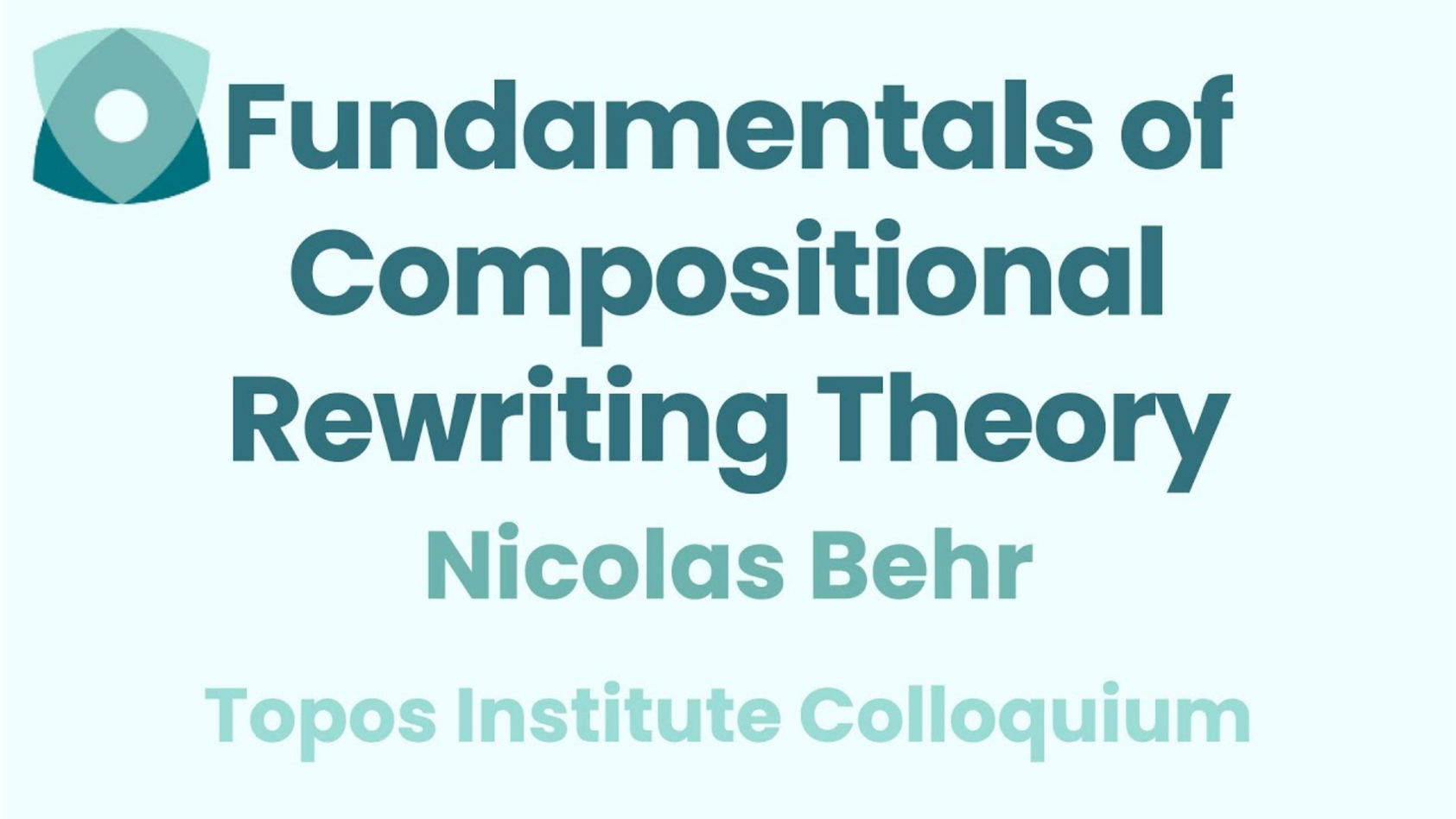
Coq-based Rewriting: towards Executable Applied Category Theory

Consortium: IRIF (UP), LIP (ENS-Lyon), LIX (École Polytechnique), Sophia-Antipolis (Inria)

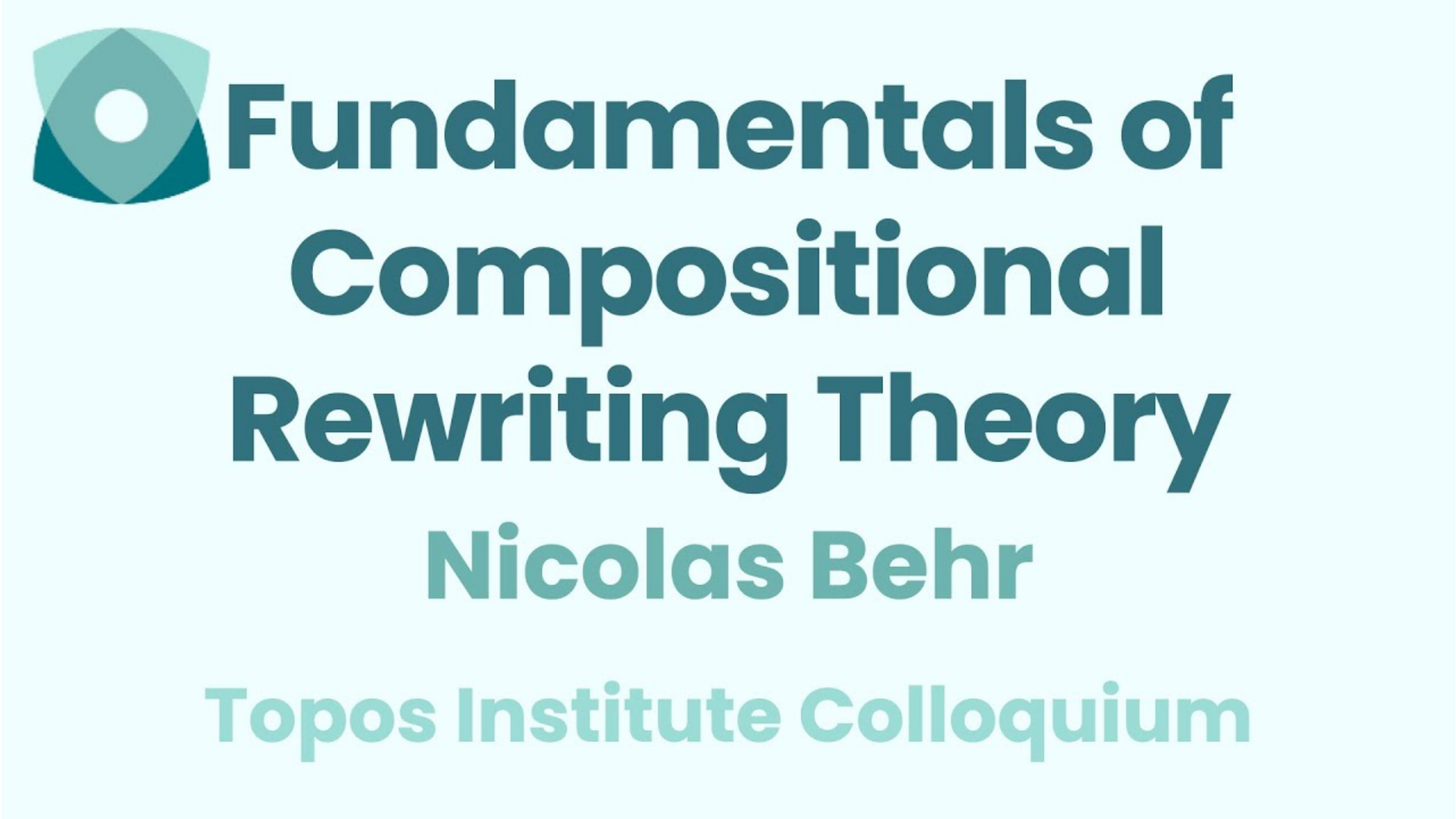


Merci beaucoup !

Appendix: some examples of **diagrammatic proofs** in **compositional rewriting theory** (cf. YouTube video for further details)

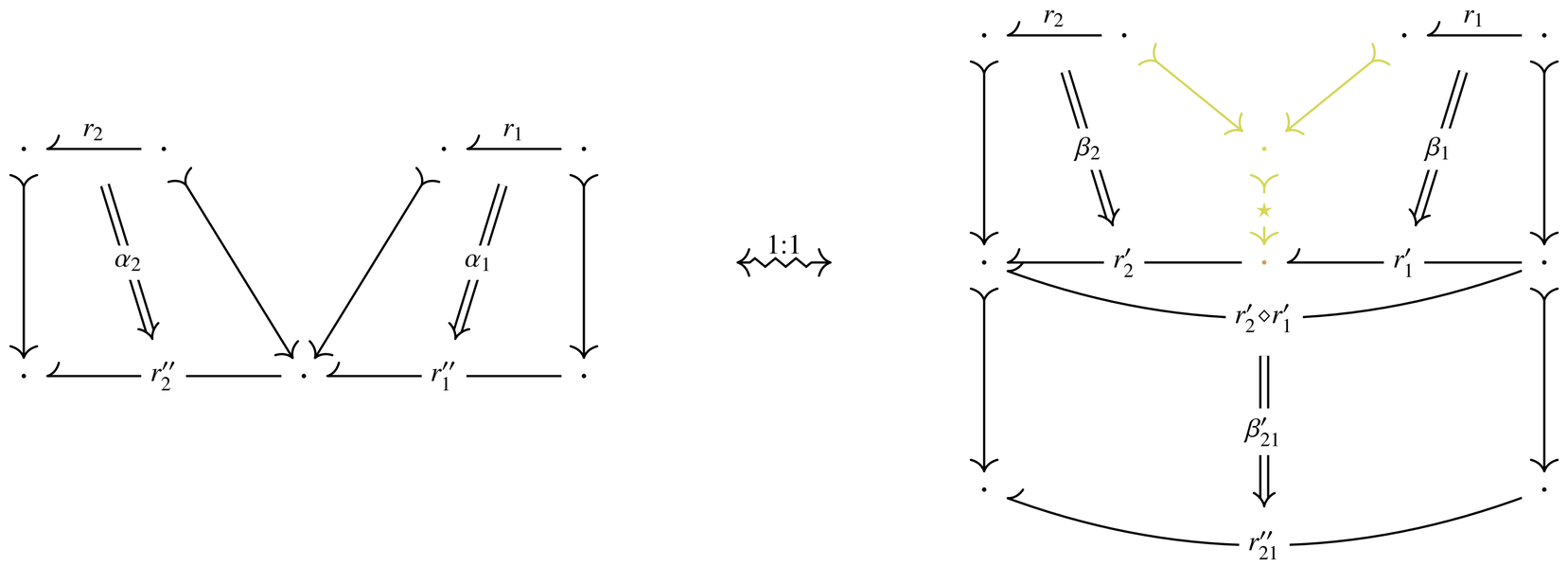


Appendix: some examples of **diagrammatic proofs** in **compositional rewriting theory** (cf. YouTube video for further details)



crDCs satisfy a (*universal!*) **Concurrency Theorem**

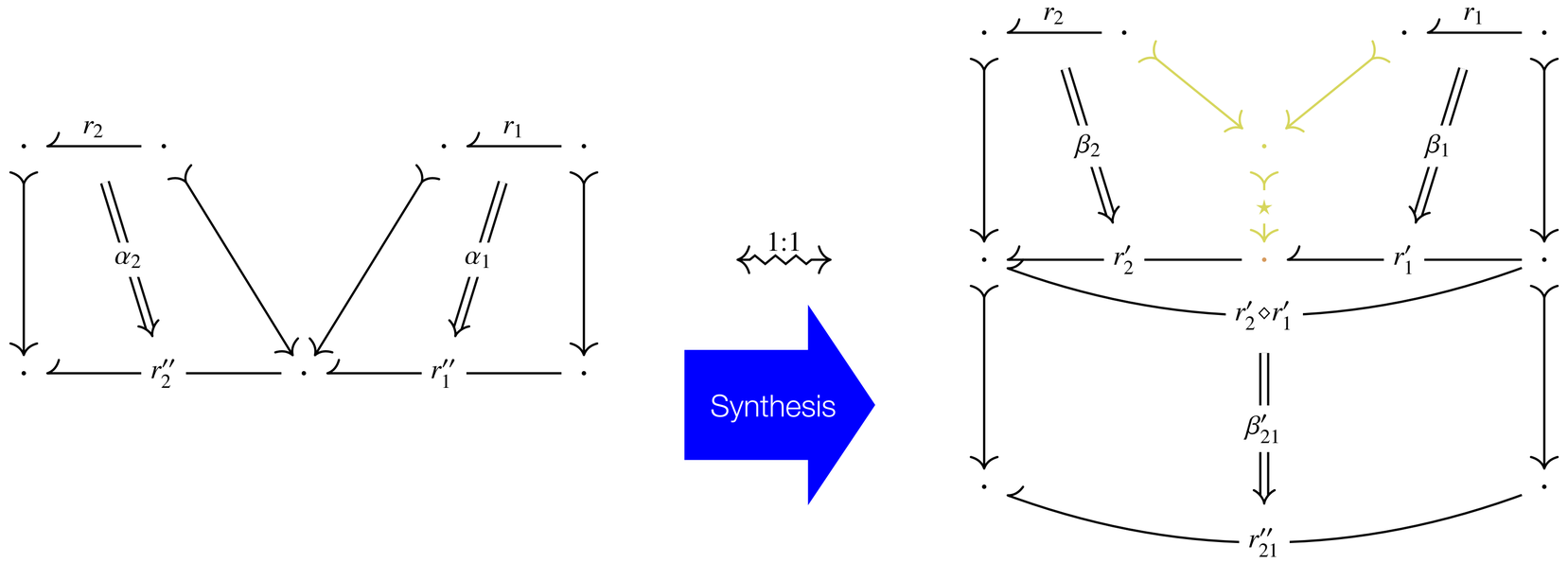
Theorem 8. *Let $\mathbb{D}$ be a compositional rewriting double category. Then the following statements hold (where the morphism marked $\star$ in the diagram on the right is a residue, and the cospan into its domain a multi-sum element.):*



(44)

crDCs satisfy a (*universal!*) **Concurrency Theorem**

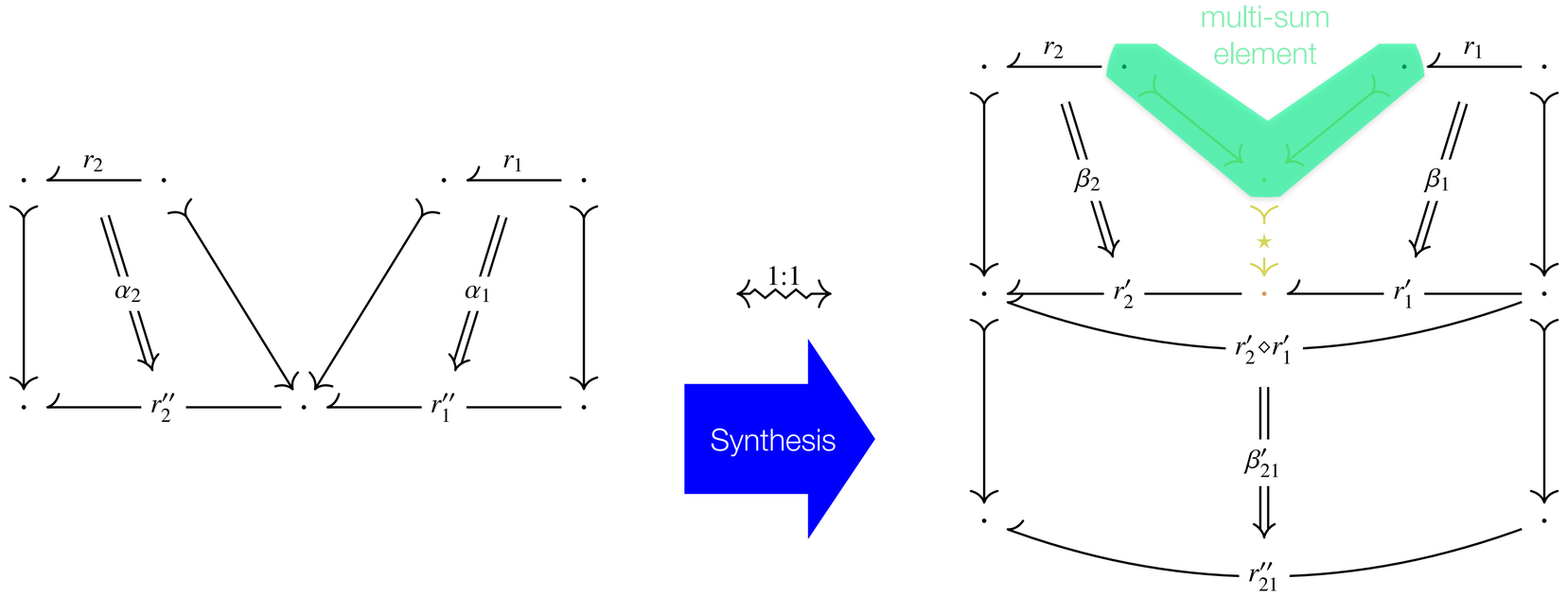
Theorem 8. *Let $\mathbb{D}$ be a compositional rewriting double category. Then the following statements hold (where the morphism marked $\star$ in the diagram on the right is a residue, and the cospan into its domain a multi-sum element.):*



(44)

crDCs satisfy a (*universal!*) **Concurrency Theorem**

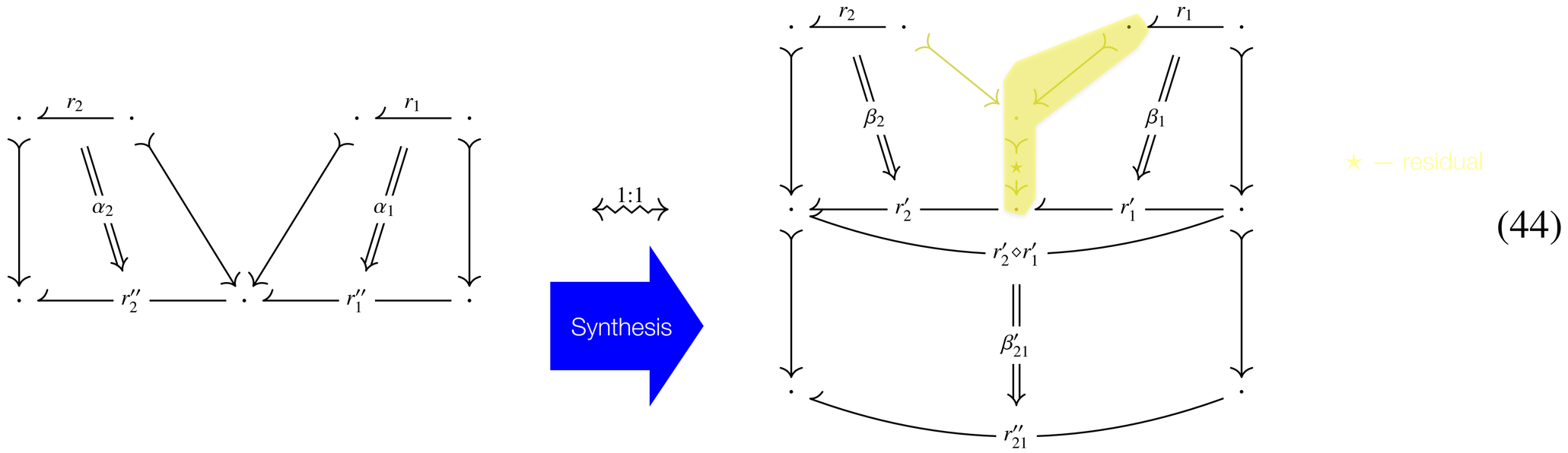
Theorem 8. *Let $\mathbb{D}$ be a compositional rewriting double category. Then the following statements hold (where the morphism marked $\star$ in the diagram on the right is a residue, and the cospan into its domain a multi-sum element.):*



(44)

crDCs satisfy a (*universal!*) **Concurrency Theorem**

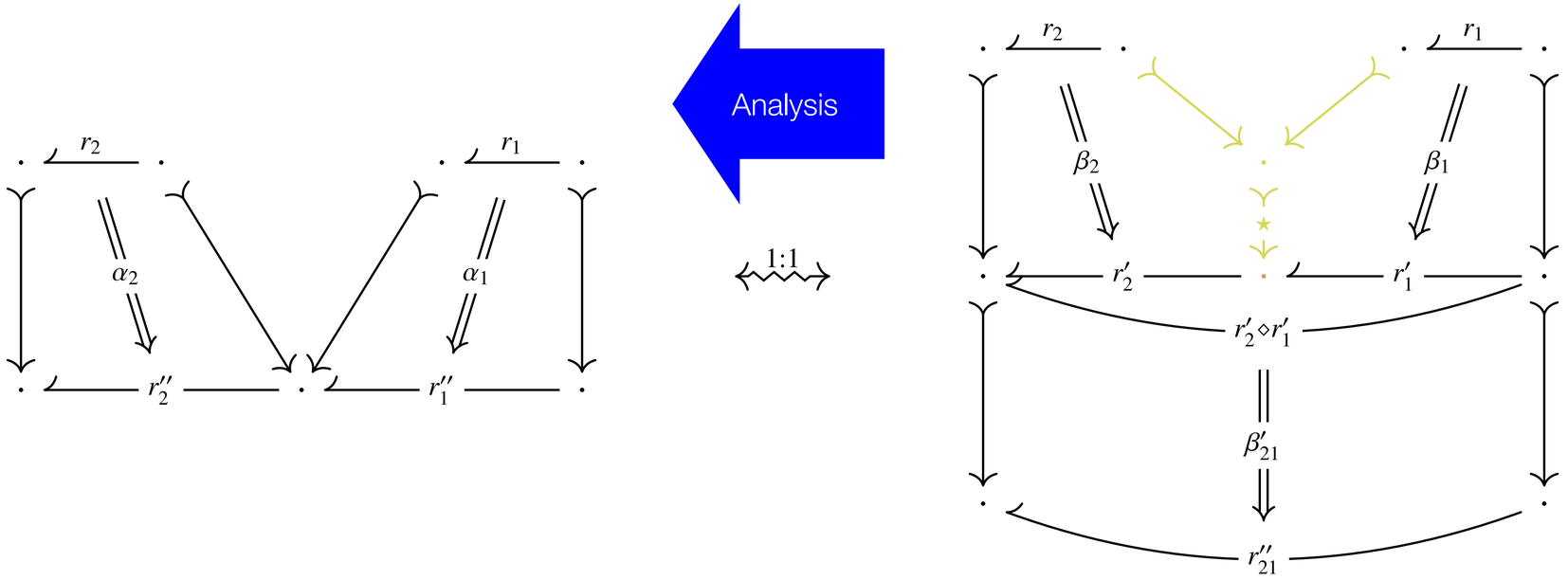
Theorem 8. *Let $\mathbb{D}$ be a compositional rewriting double category. Then the following statements hold (where the morphism marked $\star$ in the diagram on the right is a residue, and the cospan into its domain a multi-sum element.):*



(44)

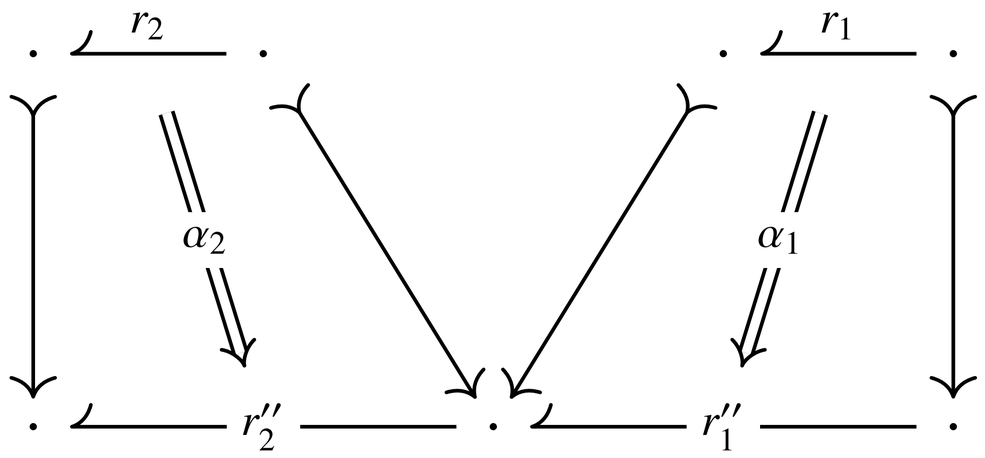
crDCs satisfy a (*universal!*) **Concurrency Theorem**

Theorem 8. *Let $\mathbb{D}$ be a compositional rewriting double category. Then the following statements hold (where the morphism marked $\star$ in the diagram on the right is a residue, and the cospan into its domain a multi-sum element.):*



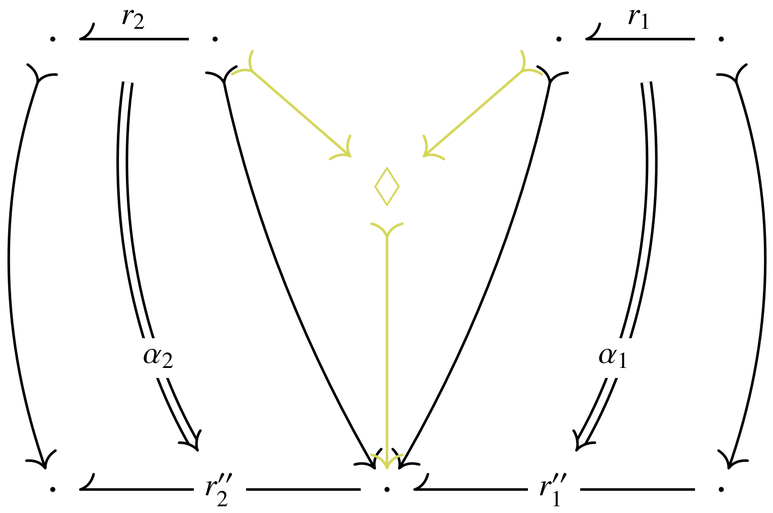
(44)

crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



PROOF. **Synthesis part:** Construct the diagram in (45) from the premise as follows:

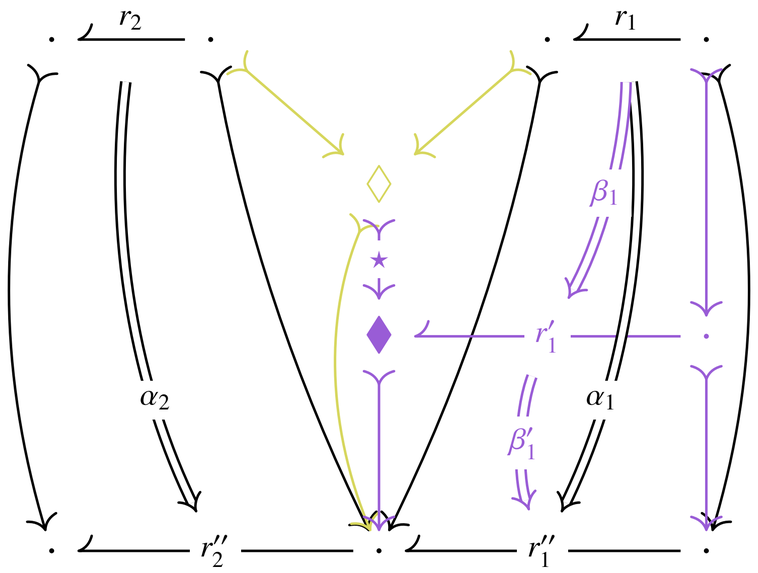
crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



PROOF. **Synthesis part:** Construct the diagram in (45) from the premise as follows:

- Via the universal property of multi-sums, there exists a cospan of $\mathbb{D}_0$-morphisms into an object $\Diamond$ and a mediating $\mathcal{M}$-morphism $\Diamond \rightarrowtail \cdot$.

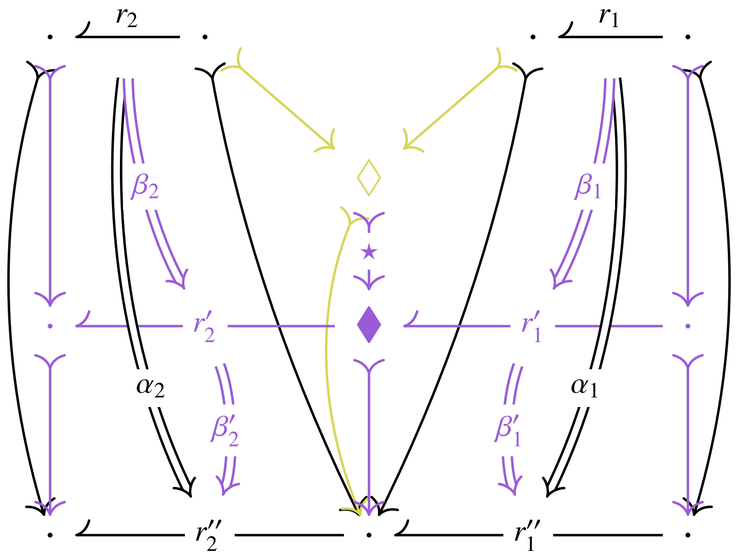
crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



PROOF. **Synthesis part:** Construct the diagram in (45) from the premise as follows:

- Via the universal property of multi-sums, there exists a cospan of $\mathbb{D}_0$-morphisms into an object $\Diamond$ and a mediating $\mathcal{M}$-morphism $\Diamond \rightarrowtail \cdot$.
- Since the target functor $T : \mathbb{D}_1 \to \mathbb{D}_0$ is a residual multi-opfibration, there exists a residue $\Diamond \rightarrowtail \blacklozenge$ (marked $\star$) and an $\mathbb{D}_0$-morphism $\blacklozenge \rightarrowtail \cdot$ such that $\alpha_1 = \beta_1' \diamond_v \beta_1$.

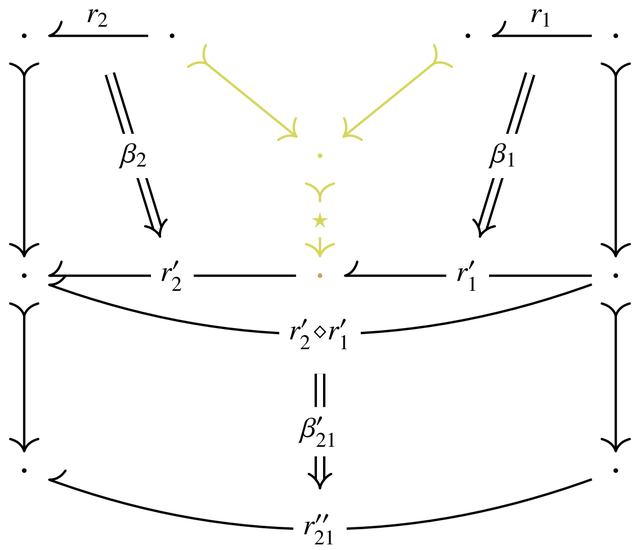
crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



PROOF. **Synthesis part:** Construct the diagram in (45) from the premise as follows:

- Via the universal property of multi-sums, there exists a cospan of $\mathbb{D}_0$-morphisms into an object $\Diamond$ and a mediating $\mathcal{M}$-morphism $\Diamond \rightarrowtail \cdot$.
- Since the target functor $T : \mathbb{D}_1 \to \mathbb{D}_0$ is a residual multi-opfibration, there exists a residue $\Diamond \rightarrowtail \blacklozenge$ (marked $\star$) and an $\mathbb{D}_0$-morphism $\blacklozenge \rightarrowtail \cdot$ such that $\alpha_1 = \beta'_1 \diamond_v \beta_1$.
- Since the source functor $S : \mathbb{D}_1 \to \mathbb{D}_0$ is a multi-opfibration, there exist direct derivations β_2 and β'_2 such that $\alpha_2 = \beta'_2 \diamond_v \beta_2$. Thus the claim follows by letting $\beta_{21} := \beta'_2 \diamond_h \beta'_1$.

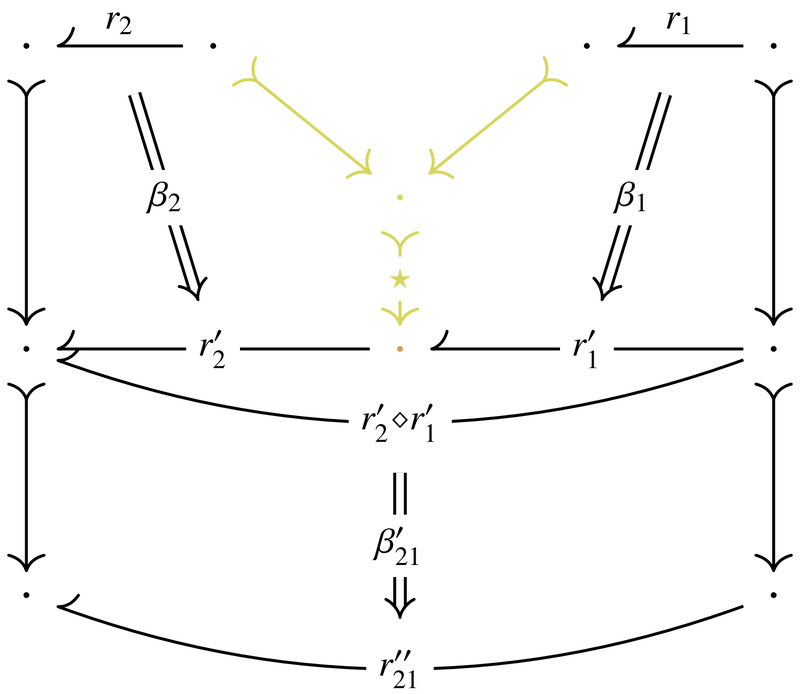
crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



PROOF. **Synthesis part:** Construct the diagram in (45) from the premise as follows:

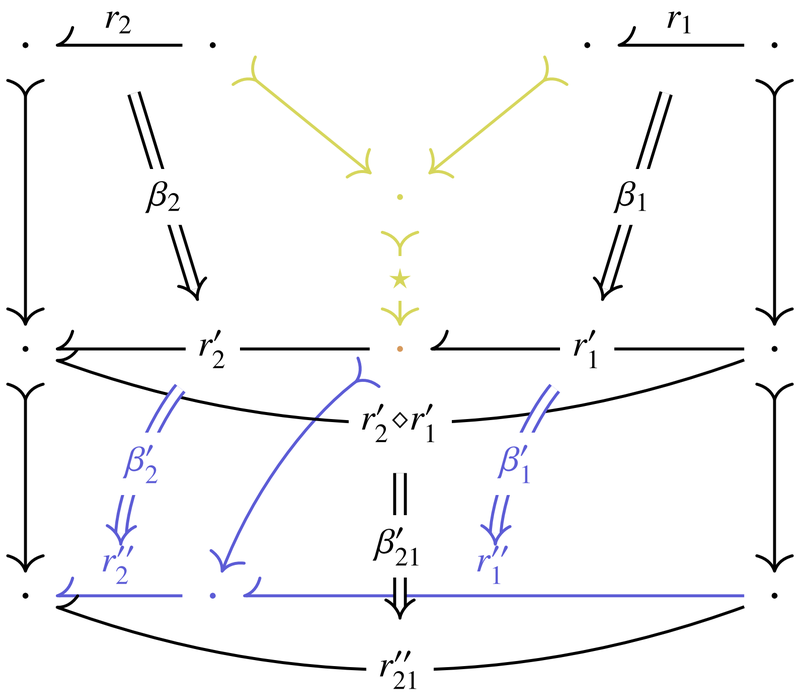
- Via the universal property of multi-sums, there exists a cospan of $\mathbb{D}_0$-morphisms into an object $\Diamond$ and a mediating $\mathcal{M}$-morphism $\Diamond \rightarrowtail \cdot$.
- Since the target functor $T : \mathbb{D}_1 \to \mathbb{D}_0$ is a residual multi-opfibration, there exists a residue $\Diamond \rightarrowtail \blacklozenge$ (marked $\star$) and an $\mathbb{D}_0$-morphism $\blacklozenge \rightarrowtail \cdot$ such that $\alpha_1 = \beta'_1 \diamond_v \beta_1$.
- Since the source functor $S : \mathbb{D}_1 \to \mathbb{D}_0$ is a multi-opfibration, there exist direct derivations β_2 and β'_2 such that $\alpha_2 = \beta'_2 \diamond_v \beta_2$. Thus the claim follows by letting $\beta_{21} := \beta'_2 \diamond_h \beta'_1$.

crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



Analysis part: Construct the diagram in (46) as follows:

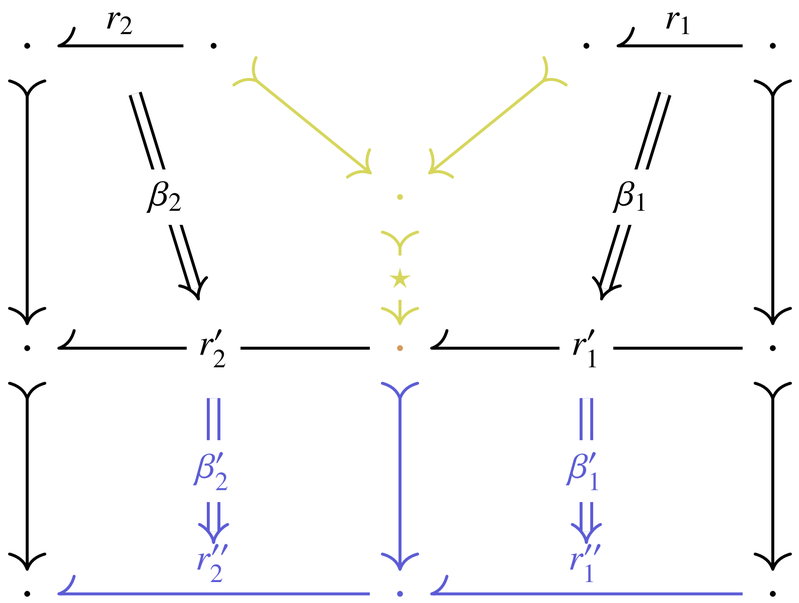
crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



Analysis part: Construct the diagram in (46) as follows:

- By the horizontal decomposition property of squares in $\mathbb{D}$, there exist squares β'_2 and β'_1 such that $\beta_{21} = \beta'_2 \diamond_h \beta'_1$.

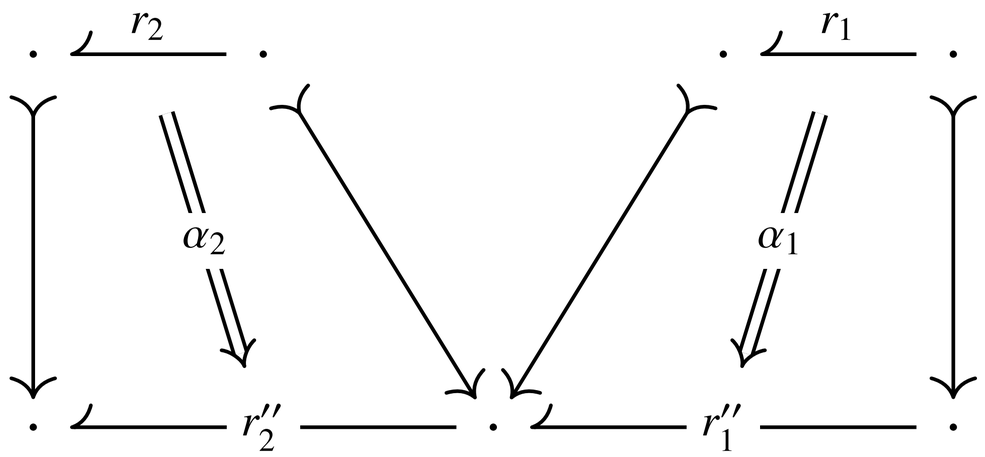
crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



Analysis part: Construct the diagram in (46) as follows:

- By the horizontal decomposition property of squares in $\mathbb{D}$, there exist squares β'_2 and β'_1 such that $\beta_{21} = \beta'_2 \diamond_h \beta'_1$.
- The claim follows be letting $\alpha_i := \beta'_i \diamond_v \beta_i$ for $i = 1, 2$.

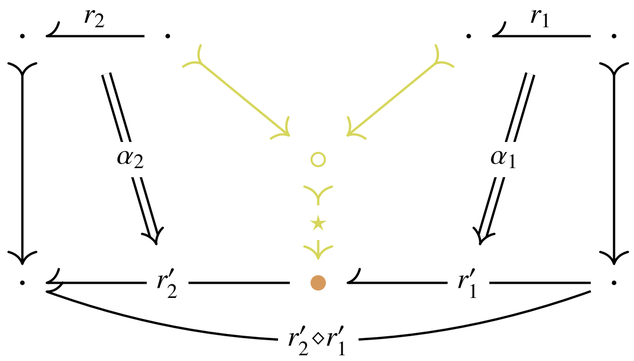
crDCs satisfy a (*universal!*) **Concurrency Theorem — PROOF**



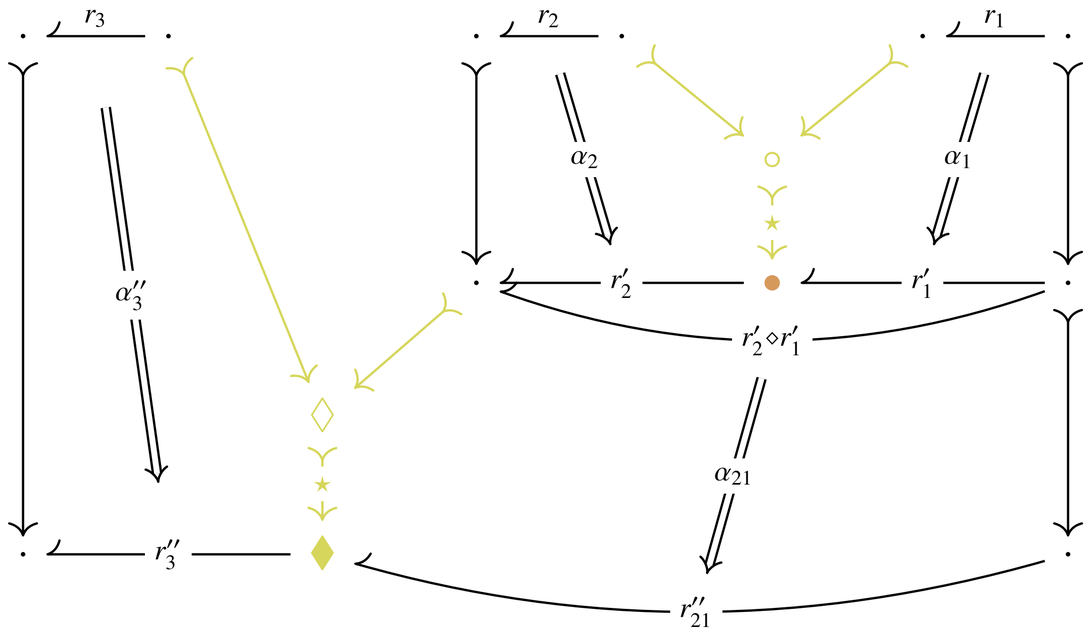
Analysis part: Construct the diagram in (46) as follows:

- By the horizontal decomposition property of squares in $\mathbb{D}$, there exist squares β_2' and β_1' such that $\beta_{21} = \beta_2' \diamond_h \beta_1'$.
- The claim follows be letting $\alpha_i := \beta_i' \diamond_v \beta_i$ for $i = 1, 2$.

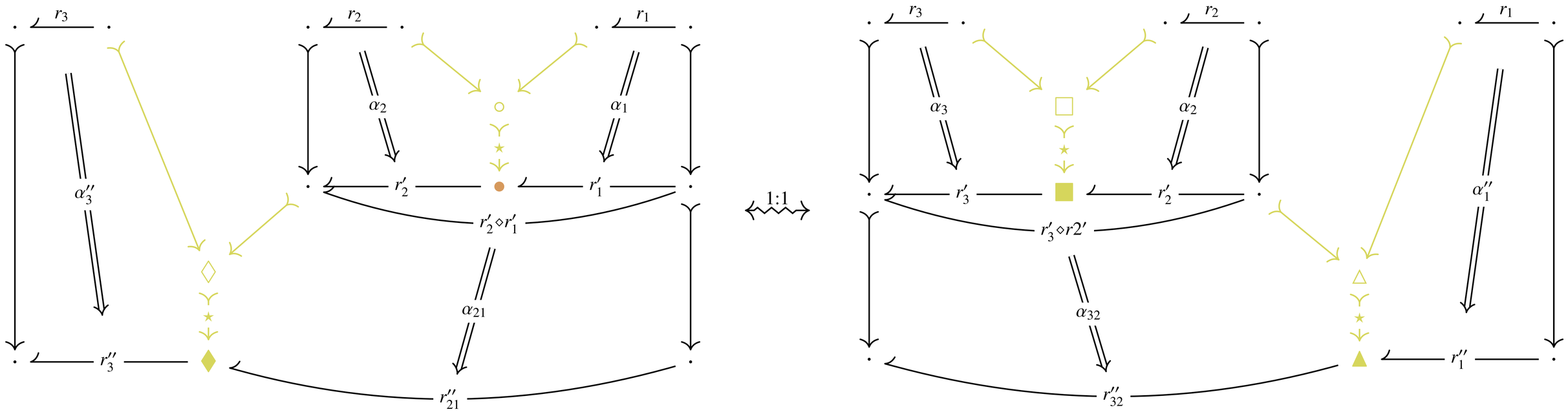
crDCs satisfy a (*universal!*) **Associativity Theorem** (= Thm. 9 in FCRT)



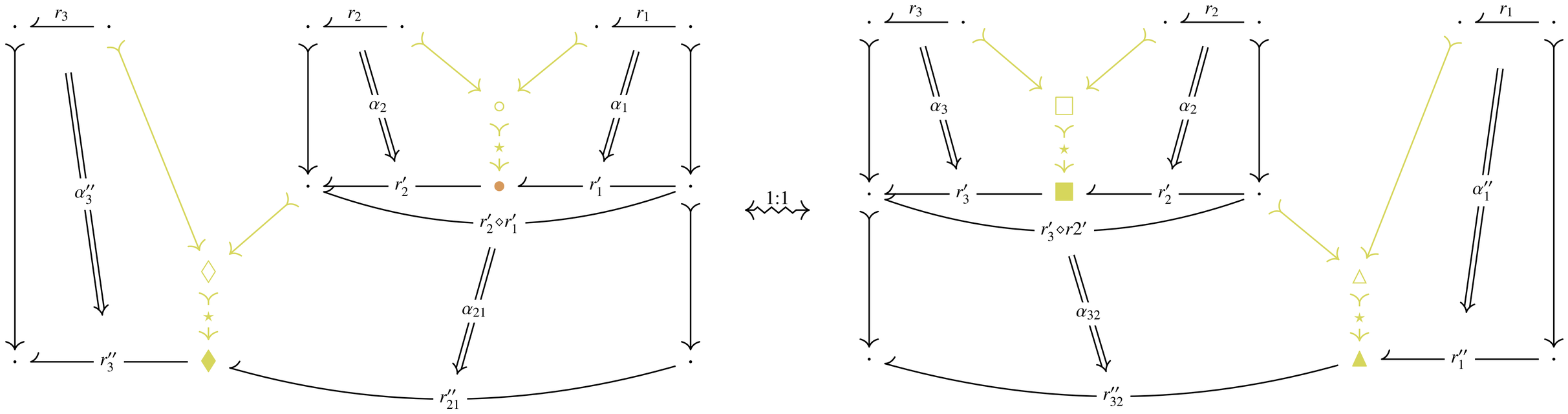
crDCs satisfy a (*universal!*) **Associativity Theorem** (= Thm. 9 in FCRT)



crDCs satisfy a (*universal!*) **Associativity Theorem** (= Thm. 9 in FCRT)



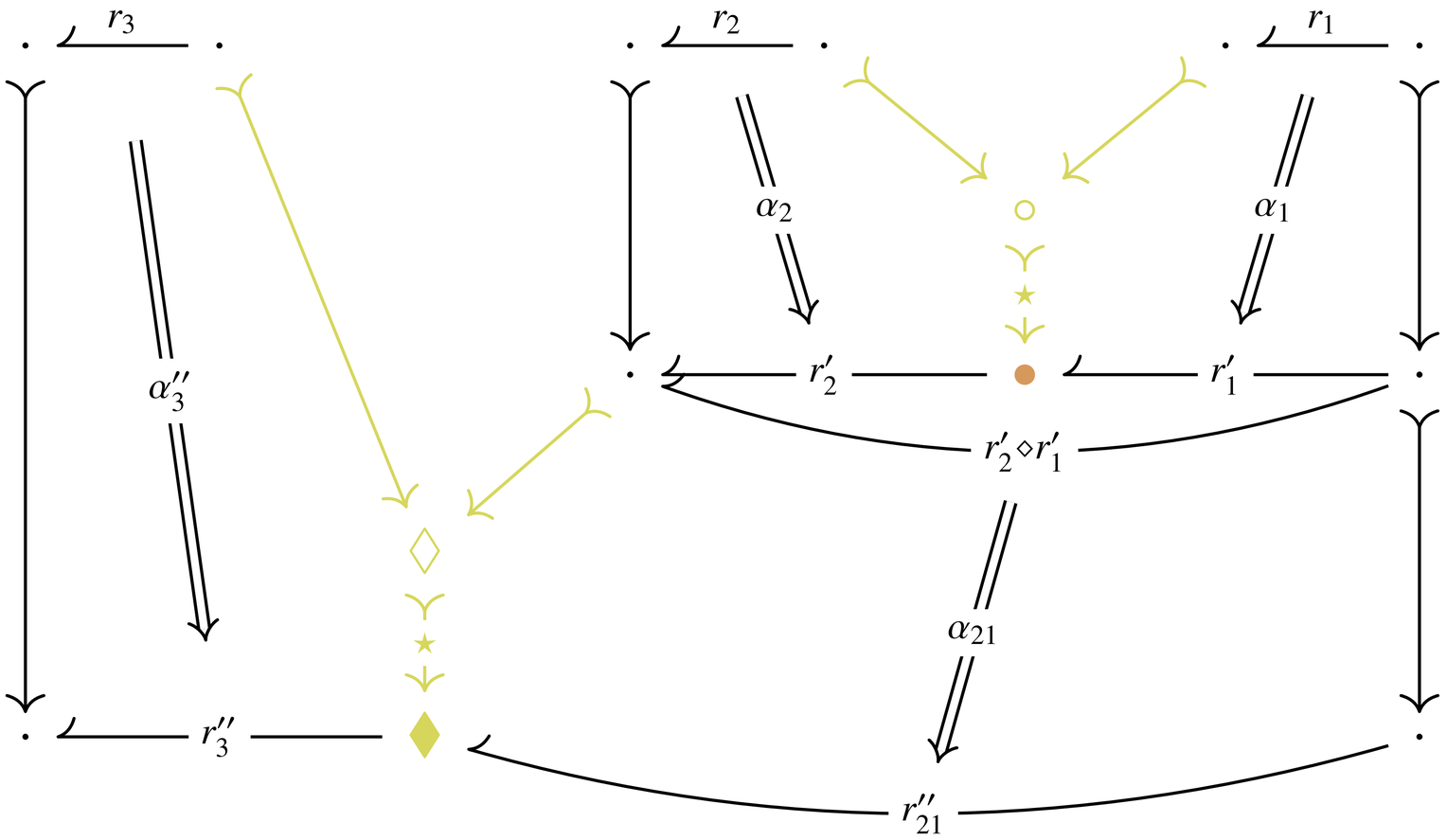
crDCs satisfy a (*universal!*) **Associativity Theorem** (= Thm. 9 in FCRT)



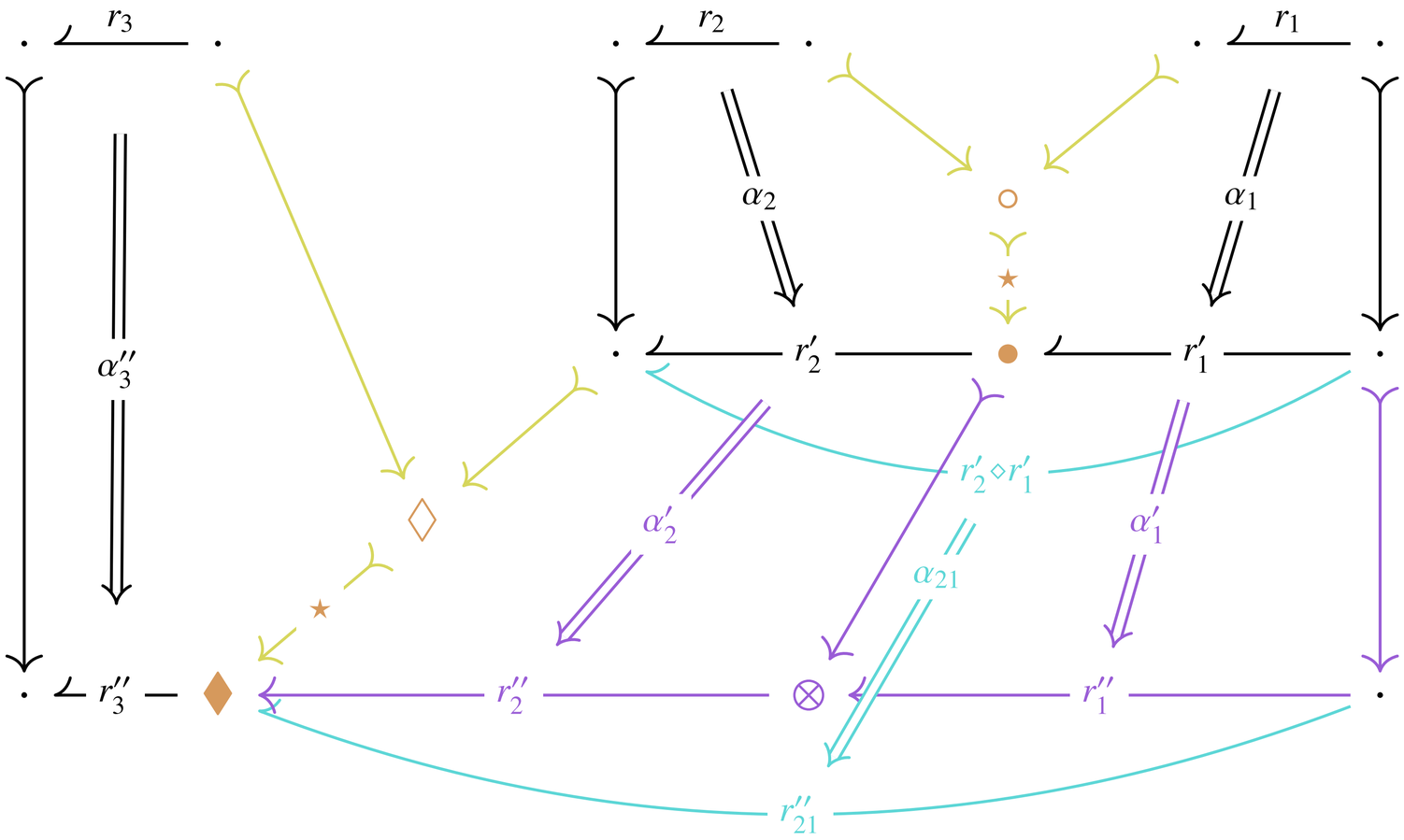
Moreover, the equivalence is such that in addition

$$r''_3 \diamond_h r''_{21} \cong r''_{32} \diamond_h r''_1 \,. \tag{49}$$

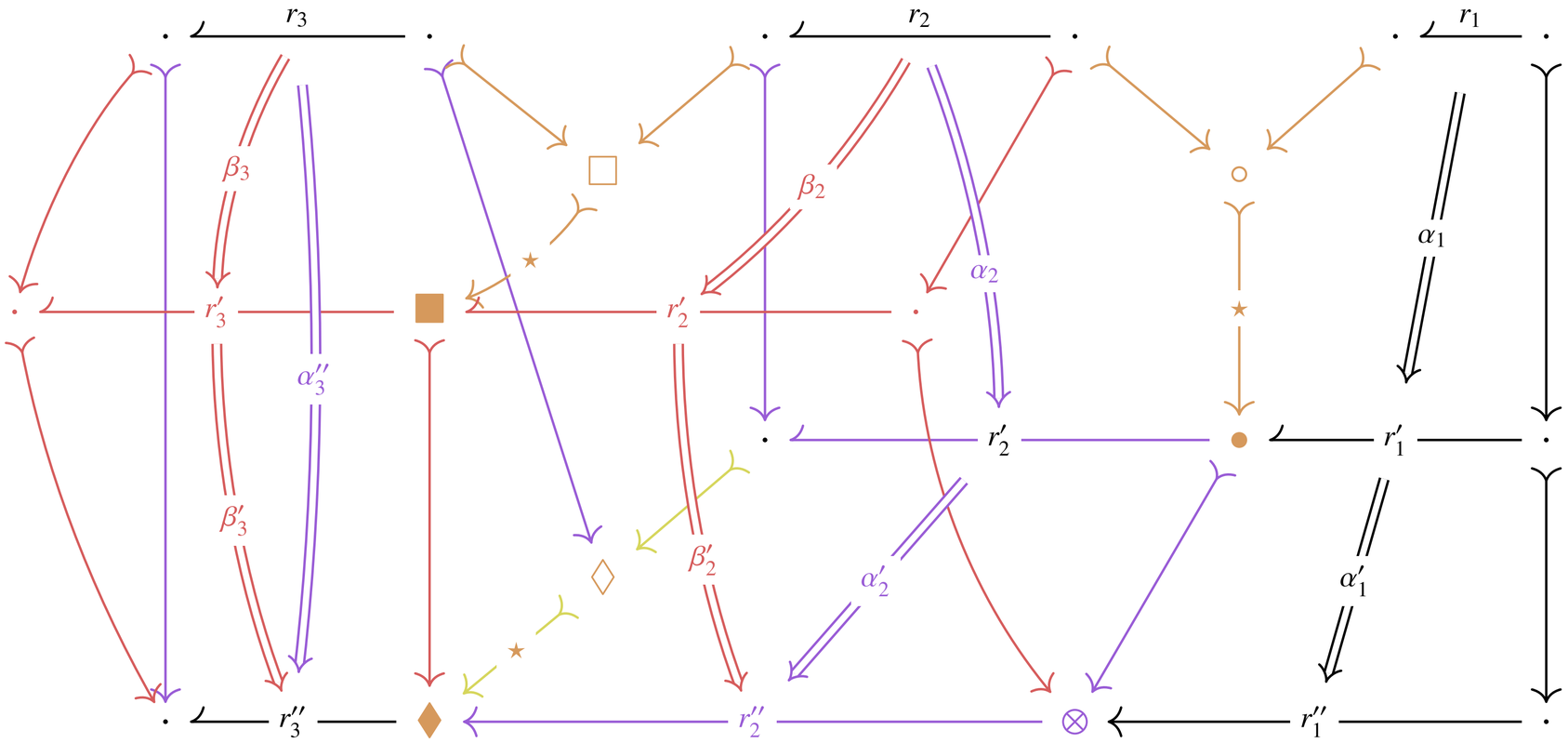
crDCs satisfy a (*universal!*) **Associativity Theorem** — PROOF SKETCH



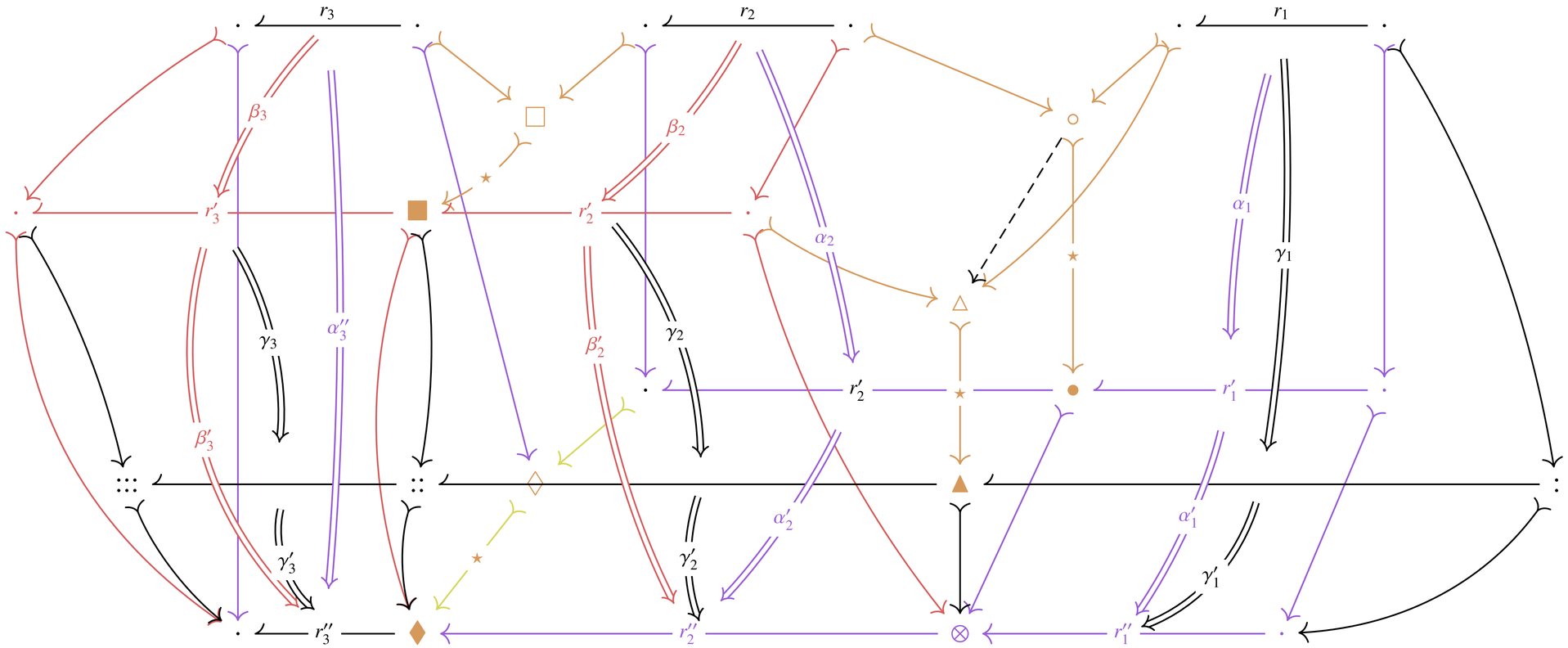
crDCs satisfy a (*universal!*) **Associativity Theorem** — PROOF SKETCH



crDCs satisfy a (*universal!*) **Associativity Theorem** — PROOF SKETCH



crDCs satisfy a (*universal!*) **Associativity Theorem** — PROOF SKETCH



crDCs satisfy a (*universal!*) **Associativity Theorem** — PROOF SKETCH