THE ESSENCE OF TYPE-THEORETIC ELABORATION

joint work with Andrej Bauer

Anja Petković Komel

QUICKEST INTRO TO TYPE THEORY

Type theories are deductive systems:

- mostly used in proof assistants
- we derive terms of types t : A
- types are “theorems”, terms are “proofs”
- think of a programming language with fancy typing system and rules how to derive terms

WHAT IS ELABORATION?

Thorsten Altenkirch @taooftypes · 9 Mar

Replying to @andrejbauer

Translating from a human readable notation into a core calculus by filling in inferable parts.

3

Martin Escardo @EscardoMartin · 7 Mar

Replying to @andrejbauer

Elaboration is the process of trying to automatically figure out the information that the mathematician using the proof checker left out deliberately because it is considered culturally obvious in the wide mathematical community.

14

Jon Sterling @jonmsterling · 7 Mar

Replying to @andrejbauer

It is the process of transforming things that we write down or type in (e.g. raw syntax or code) into (representations of) the mathematical objects we are talking about.

1 9

Elaboration =

Transformation into mathematical objects?

Figuring out missing (mathematical) context information?

DISCLAIMER: NOT ALL ANSWERS ARE INCLUDED

Saroupille @Saroupille · 7 Mar
Replying to @andrejbauer
Going from the user syntax to the kernel syntax? A bit similar to the compilation process actually.
1 2

Bernardo Toninho @bernpton · 7 Mar
Replying to @andrejbauer
Don't know if I qualify, but elaboration to me is going from a surface "practical" syntax (implicits / omitting type annotations in binders / universe levels) to some core fully explicit representation, which usually involves higher-order unification (but not necessarily).
1 9

Henri Tuhola @HenriTuhola · 7 Mar
Replying to @andrejbauer
It's a form of type checking where you also "elaborate" the term from one syntax into another syntax.

Brendan Zabarauskas (@brendan@types.pl) @brendanzab · 7 Mar
Replying to @andrejbauer
It's the process of expressing implicit, overloaded, context sensitive information in a surface language a simpler, more explicit core language, while checking for any mistakes in the process. Kind of like type checking, type inference, and compilation pass rolled into one.
1 9

Elaboration =

Type checking + compilation ?

DISCLAIMER: NOT ALL ANSWERS ARE INCLUDED

Esa Pulkkinen @PulkkinenEsa · 7 Mar

Replying to @andrejbauer

My understanding is elaboration refers to transition from high level of abstraction to lower one by incrementally adding missing details.
But my impression on this could be mistaken, I do still remember I was confused when I first heard of the concept.

4

???

Confused, frustrated.

We don't really know.

andgate (paranormal software detective) @the_andgate · 7 Mar

Reading papers on type theory is very frustrating, because type theorists almost never provide definitions for their esoteric terms and phrases. "Elaboration" being the most damaging example.

1

IDEA OF ELABORATION:

Adding missing information

IDEA OF ELABORATION:

- **Adding missing types**

$$\frac{\vdash A \text{ type} \quad \vdash B \text{ type} \quad x : A \vdash e : B}{\vdash \lambda(x.e) : A \to B}$$

$$\Rightarrow$$

$$\frac{\vdash A \text{ type} \quad \vdash B \text{ type} \quad x : A \vdash e : B}{\vdash \lambda(A, B, x.e) : A \to B}$$

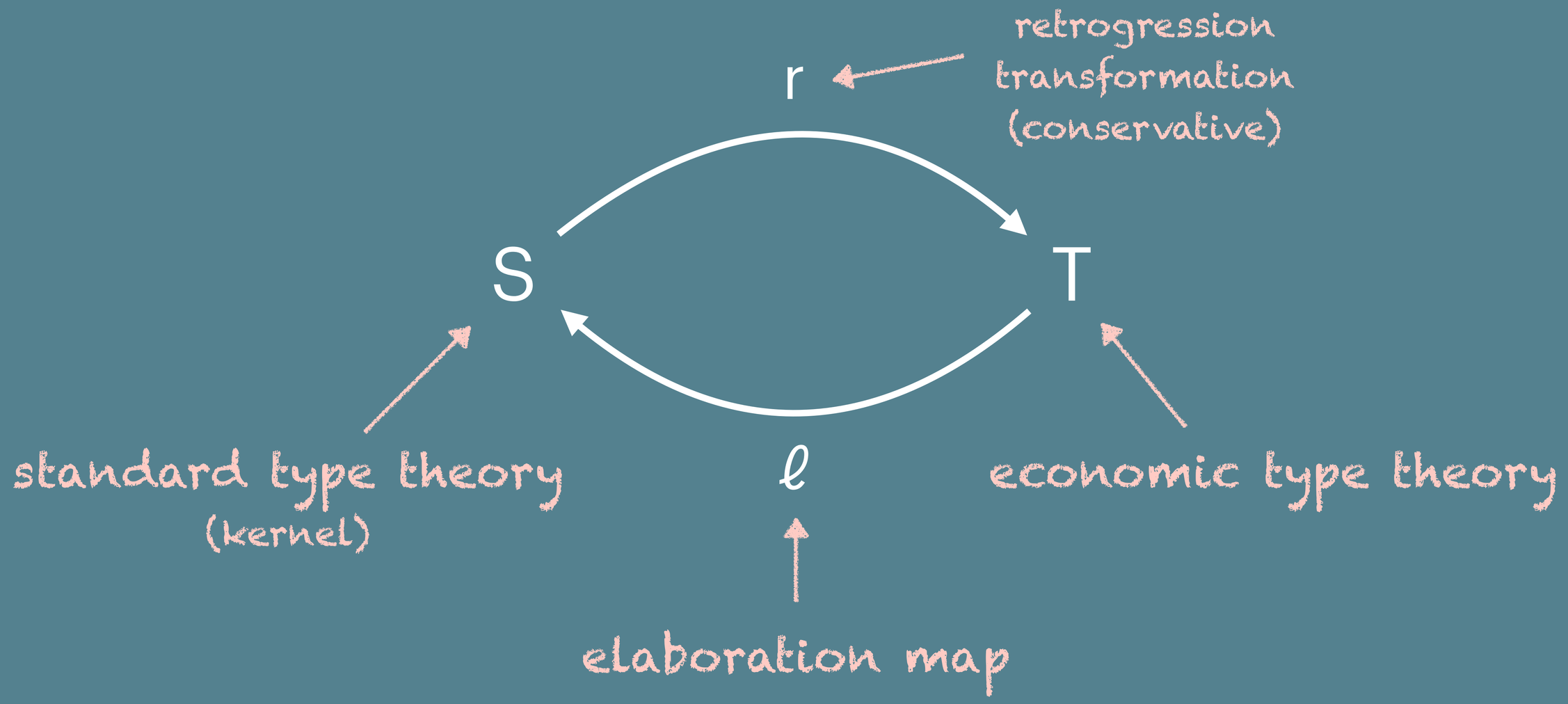
IDEA OF ELABORATION:

- **Adding missing evidence**

$$\frac{\vdash A : \mathrm{Type} \qquad \vdash B : \mathrm{Type}}{\vdash A \to B : \mathrm{Type}} \quad \Rightarrow \quad \frac{\vdash A : \mathrm{Type\_i} \qquad \vdash B : \mathrm{Type\_k}}{\vdash A \to B : \mathrm{Type\_max(i,k)}}$$

(universe levels, termination checking)

THE ESSENCE OF ELABORATION



TYPE THEORY*: INGREDIENTS

- A **signature** of symbols.
- 4 kinds of judgements.

$A \ \mathrm{type}$ $\quad a : A$ $\quad A \equiv B$ $\quad a \equiv b : A$

- Hypothetical judgements

variable context +
metavariable context $\longrightarrow$ $\Gamma \vdash \mathcal{J}$

*Type theory = Finitary type theory in the sense of Haselwarter and Bauer.

TYPE THEORY : INGREDIENTS

- **Structural rules**: Variable rule, reflexivity, symmetry and transitivity of equations etc.
- **Specific rules**:

Object rules

$$\frac{\vdash A \text{ type} \quad \vdash B \text{ type}}{\vdash A \to B \text{ type}}$$

$$\frac{\vdash A \text{ type} \quad \vdash B \text{ type} \quad x : A \vdash e : B}{\vdash \lambda(x.e) : A \to B}$$

Equality rules

$$\frac{}{\vdash N \equiv \mathbb{N}}$$

$$\frac{\vdash A \text{ type} \quad \vdash B \text{ type} \quad \vdash a : A \quad \vdash b : B}{\vdash \mathrm{fst}(\mathrm{pair}(a,b)) \equiv a : A}$$

- **Congurence rules** (for every object rule).

Such that the rules are well-formed (presuppositivity)

SYMBOL RULES

Compare the two rules.

$$\frac{\vdash A \text{ type} \quad \vdash B \text{ type} \quad \vdash p : A \times B}{\vdash \mathrm{fst}(p) : A}$$

Better for user input.

$$\frac{\vdash A \text{ type} \quad \vdash B \text{ type} \quad \vdash p : A \times B}{\vdash \mathrm{fst}(A, B, p) : A}$$

Faithfully records the (proof-relevant parts of) the premises.

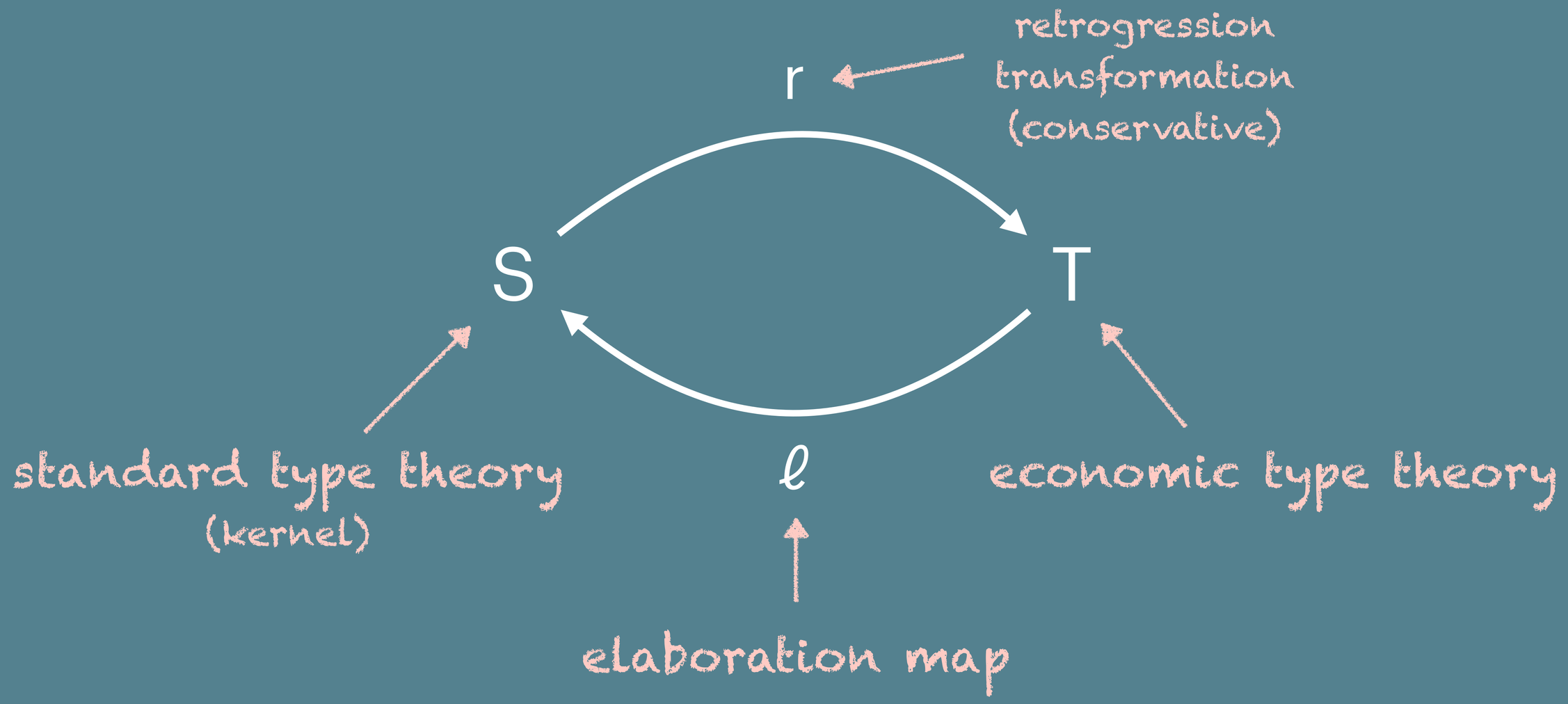
STANDARD TYPE THEORY

A type theory is **standard** if every object rule is a symbol rule and every symbol has exactly one symbol rule.

Standard type theories are **well behaved**:

- inversion
- uniqueness of typing

THE ESSENCE OF ELABORATION



TYPE-THEORETIC TRANSFORMATION

$$f : U \longrightarrow T$$

$$S \longmapsto e$$

a symbol from signature of U

a well-formed expression in T

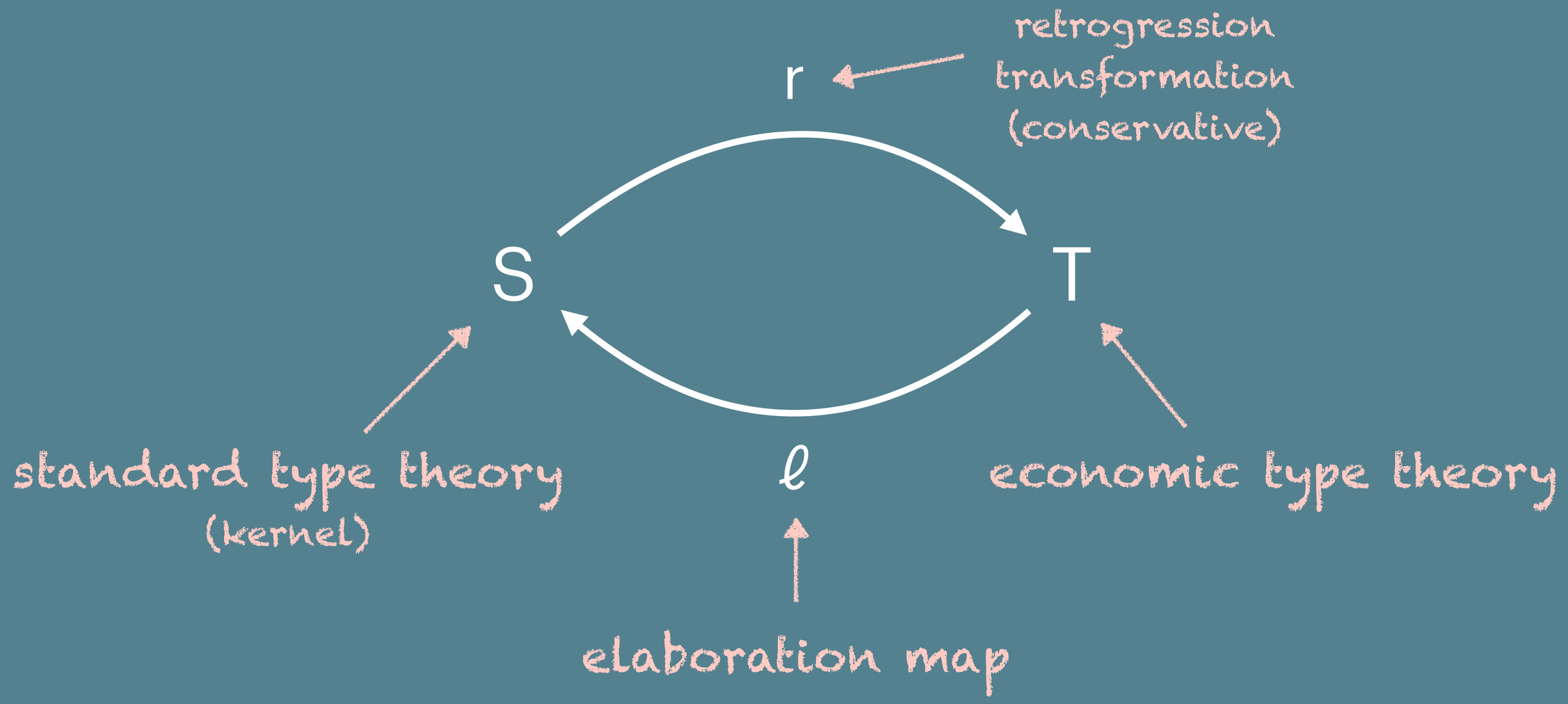
Specific rule

derivation

$$\frac{P_1 \quad \cdots \quad P_n}{\vdash \mathcal{J}} \longmapsto \frac{f_* P_1 \quad \cdots \quad f_* P_n}{\vdash f_* \mathcal{J}}$$

Transformations form a relative monad for syntax and preserve derivability.

THE ESSENCE OF ELABORATION



ELABORATION MAP

$$\begin{tikzcd} S \arrow[r, bend left, "r"] & T \arrow[l, bend left, "\ell"] \end{tikzcd}$$

$$\Gamma' \vdash \mathcal{J}' \longleftarrow\!\!\shortmid \quad \frac{\mathcal{D}}{\Gamma \vdash \mathcal{J}}$$

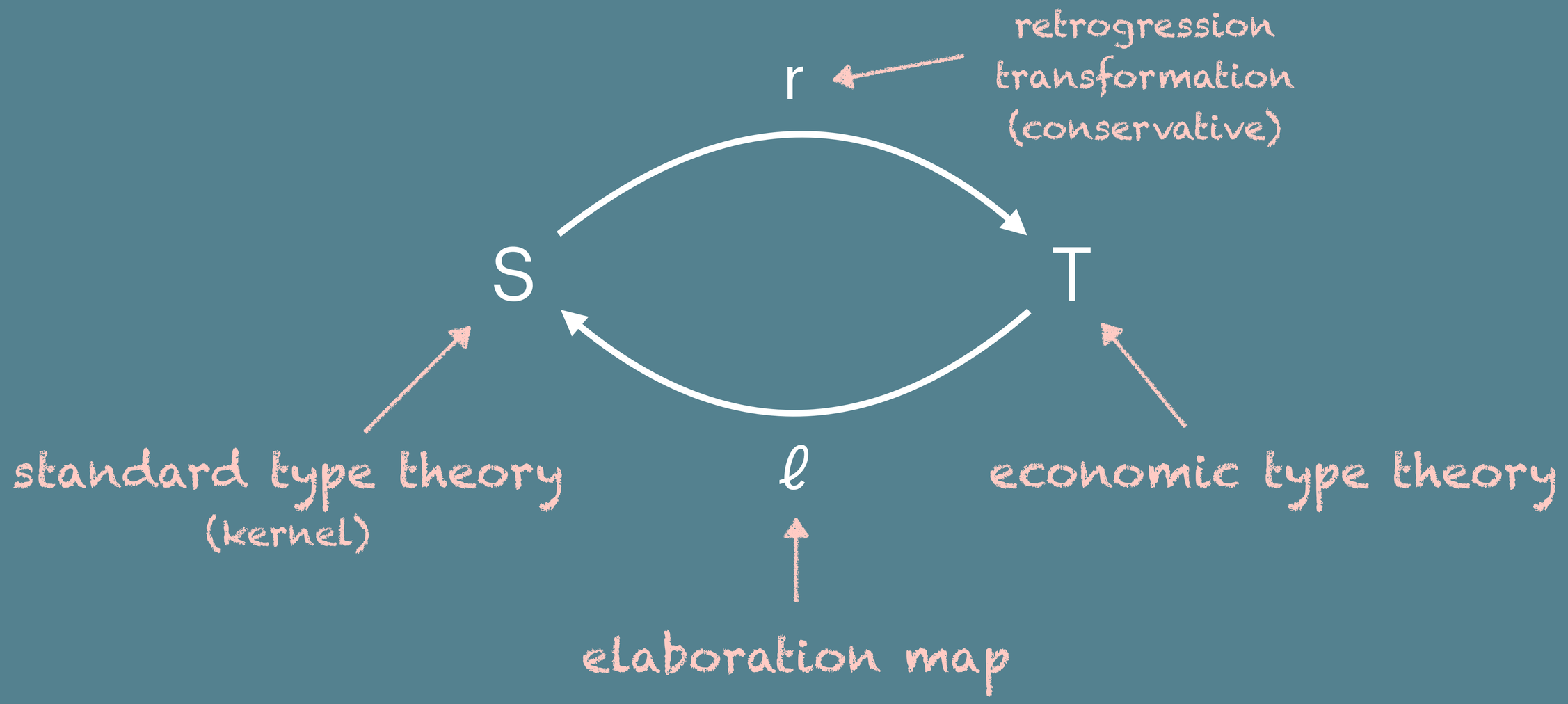
Elaboration map takes a derivation!

... such that $r_*(\Gamma' \vdash \mathcal{J}') = \Gamma \vdash \mathcal{J}$.

ℓ is a witness of surjectivity of r.

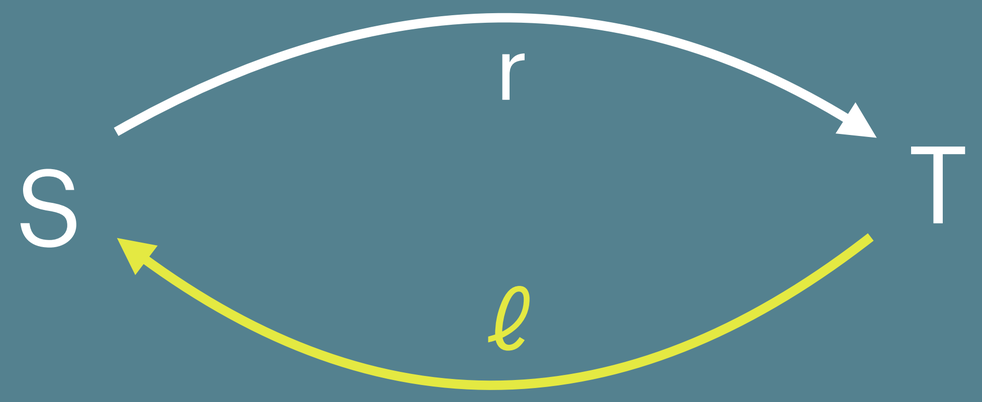
Elaboration map preserves derivability.

THE ESSENCE OF ELABORATION



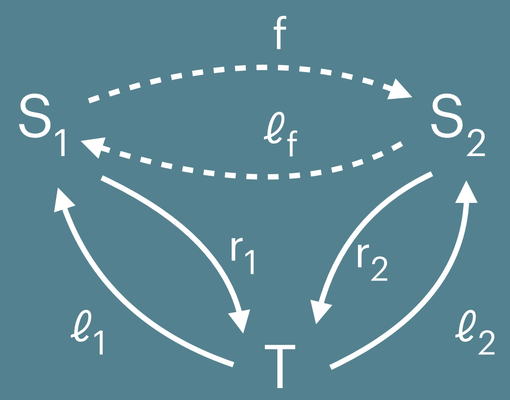
Elaboration map is unique

up-to judgemental equality.



Retrogressions are essentially the same:

they factor through each other by other retrogressions.



$$r_2 \circ f = r_1$$

f is conservative and unique up-to judgemental equality.

AN ELABORATION THEOREM

Every type theory has "an elaboration".

For every type theory T there exists a standard type theory S with a retrogression $r : S \to T$ and elaboration map $\ell : T \to S$.

ELABORATOR: ALGORITHM

ELABORATOR

Elaborator: an algorithm

in finitary type theory

takes : judgement J

outputs: a derivable elaborated judgement J' if it exists,
or reports there is none

in standard type theory

An elaborator, if it exists, is *computable* for our chosen type theory.

EXISTENCE OF ELABORATOR

T has an elaborator if and only if T has decidable type and equality checking.

Elaborator is the most general type-checking algorithm for T, if any exists.

THE ESSENCE OF ELABORATION

- **Universal property**
- **Elaboration theorem** (every economic theory can be elaborated)

$$\begin{tikzcd} S \arrow[r, bend left] & T \arrow[l, bend left, "\ell"] \arrow[l, bend right, "r"'] \end{tikzcd}$$

retrogression transformation (conservative)

standard type theory (kernel)

economic type theory

elaboration map