Performance Evaluation for Gradual Typing

Asumu Takikawa
*Ben Greenman
Jan Vitek

Daniel Feltey
Max S. New
Matthias Felleisen

Northeastern University

9 years of sound gradual typing

Gradual Typing for Functional Languages,
Jeremy Siek & Walid Taha.
SFP '06

Interlanguage Migration: From Scripts to Programs,
Sam Tobin-Hochstadt & Matthias Felleisen.
OOPSLA '06

9 years of sound gradual typing

Micro

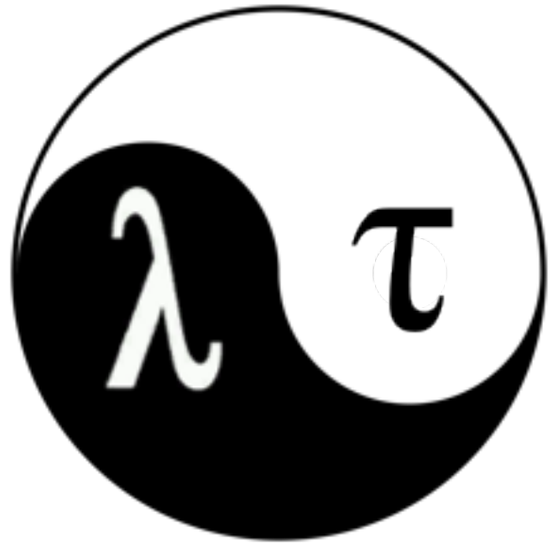
Macro

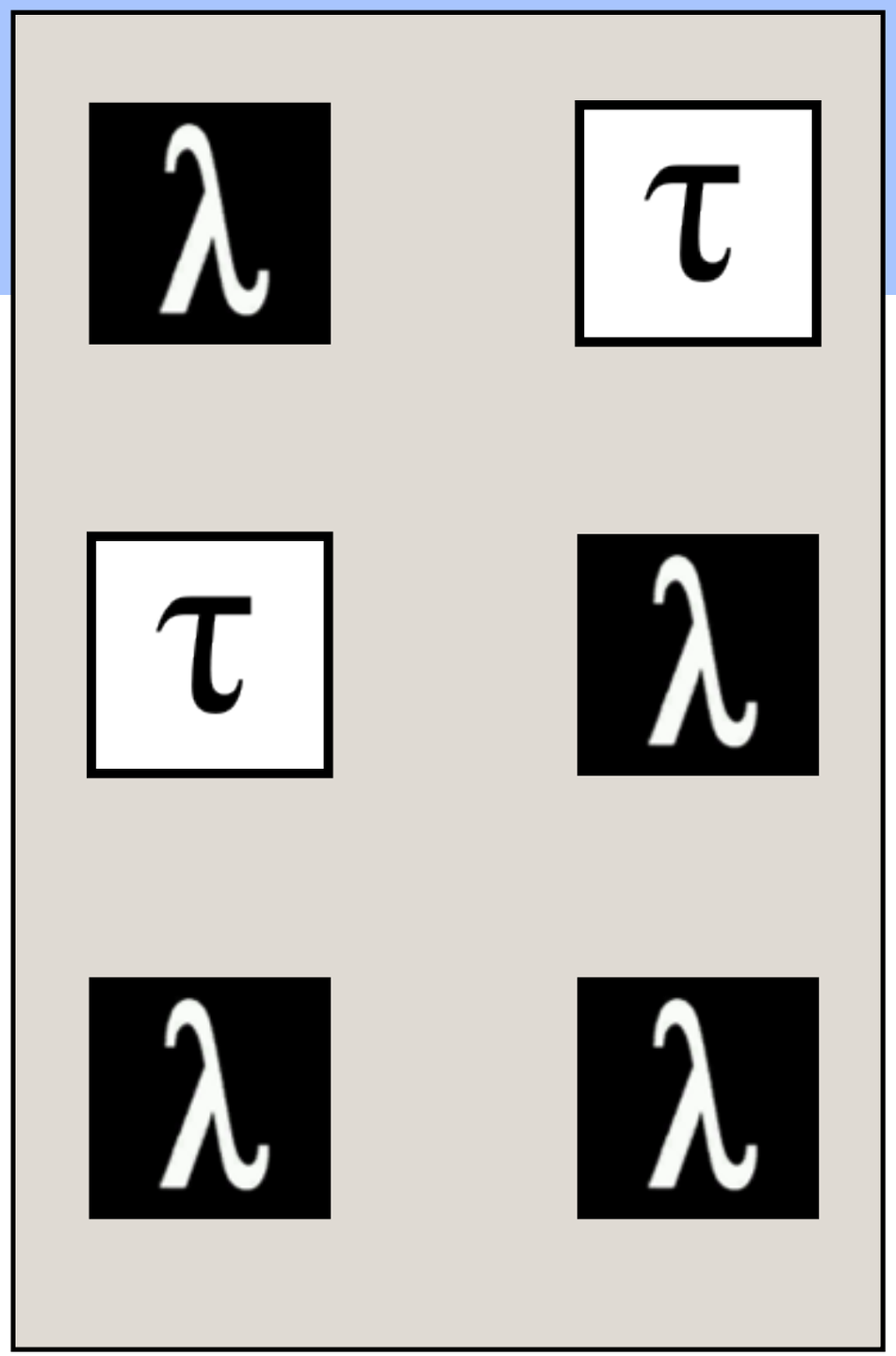
Micro

- focus on single-module programs

- type system catches incompatibilities

- all values implicitly type DYN





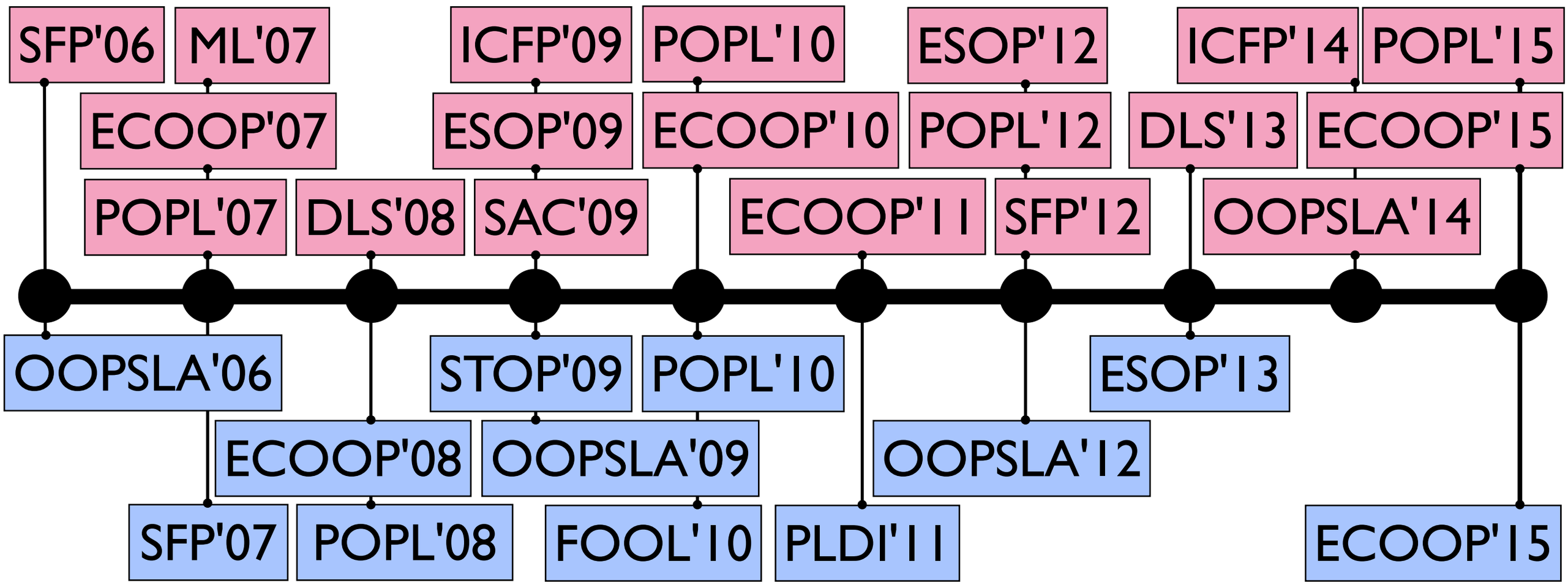
Macro

- multi-module, untyped programs

- eliminate a class of errors

- each *module* fully typed/untyped

9 years of sound gradual typing



Over **50** publications since 2006

http://github.com/samth/gradual-typing-bib

Performance?

Performance?

4x

10x

22x

33x

9 years of publications

and nobody publishes evaluations

because performance sucks

Gradual Typing ... is practical?

Gradual Typing ... is practical?

Performance? 

Soundness? 

Types without Soundness

- catch "obvious" logical errors

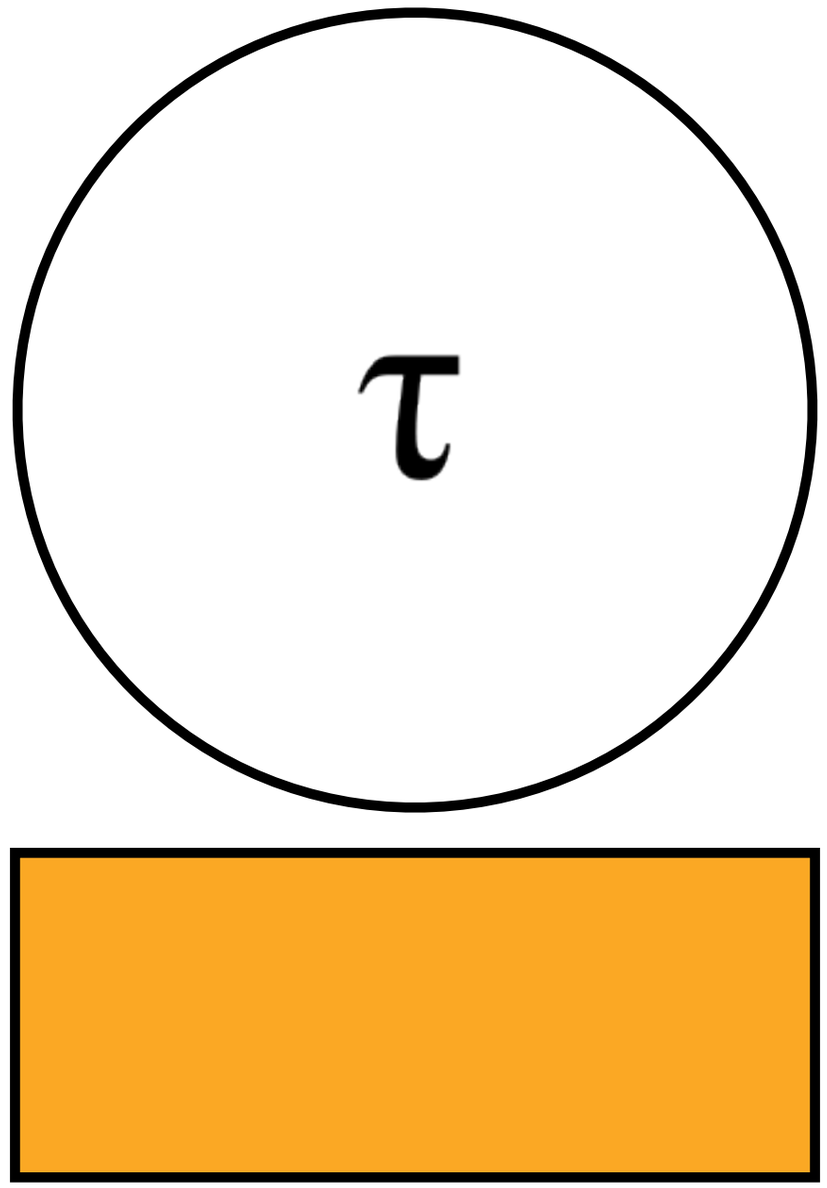
- serve as documentation

- may not be correct, or complete

Types + Soundness

eliminate a class of errors from typed code !

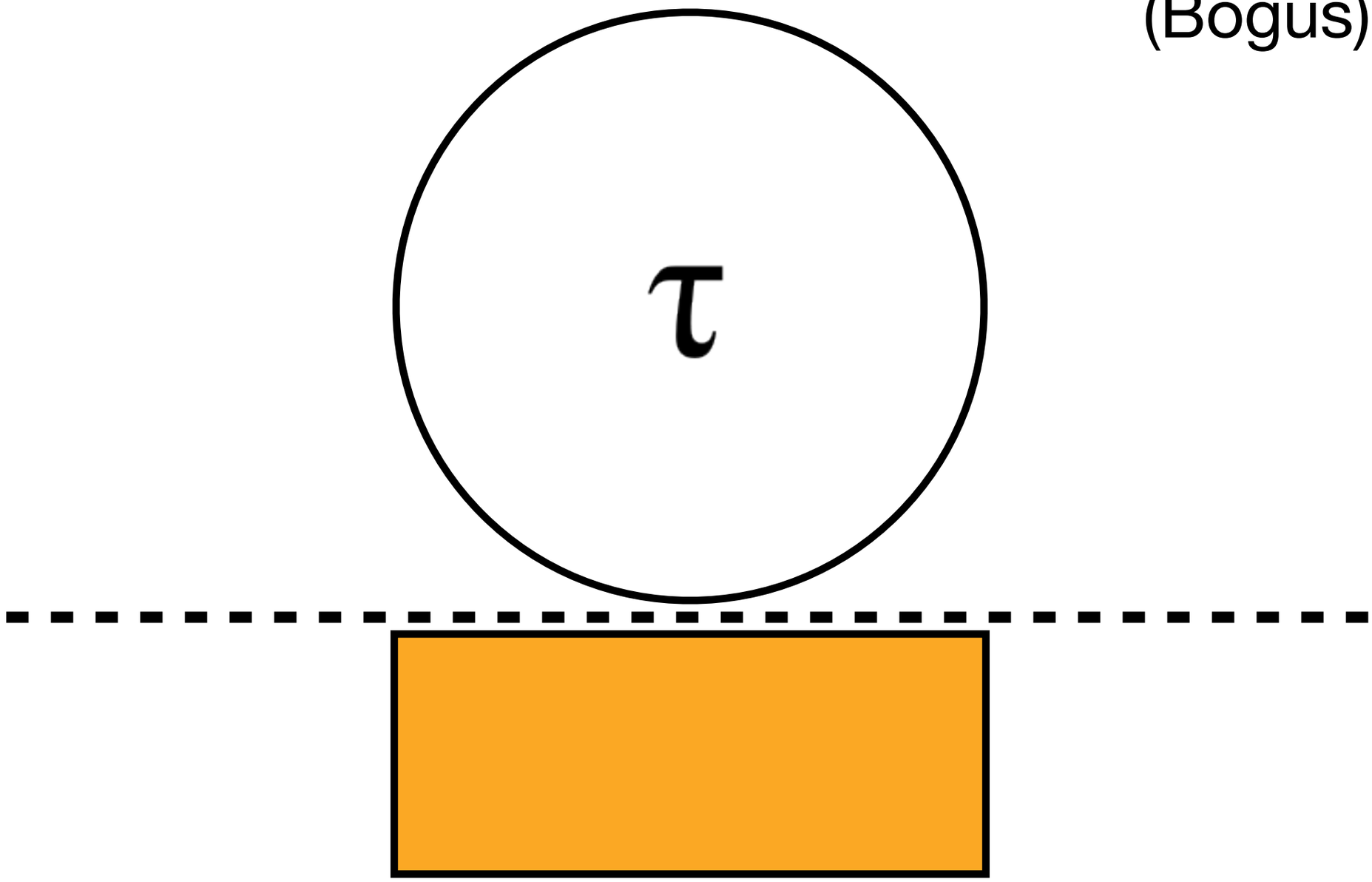
Every typed language...



Rests on an
untyped runtime

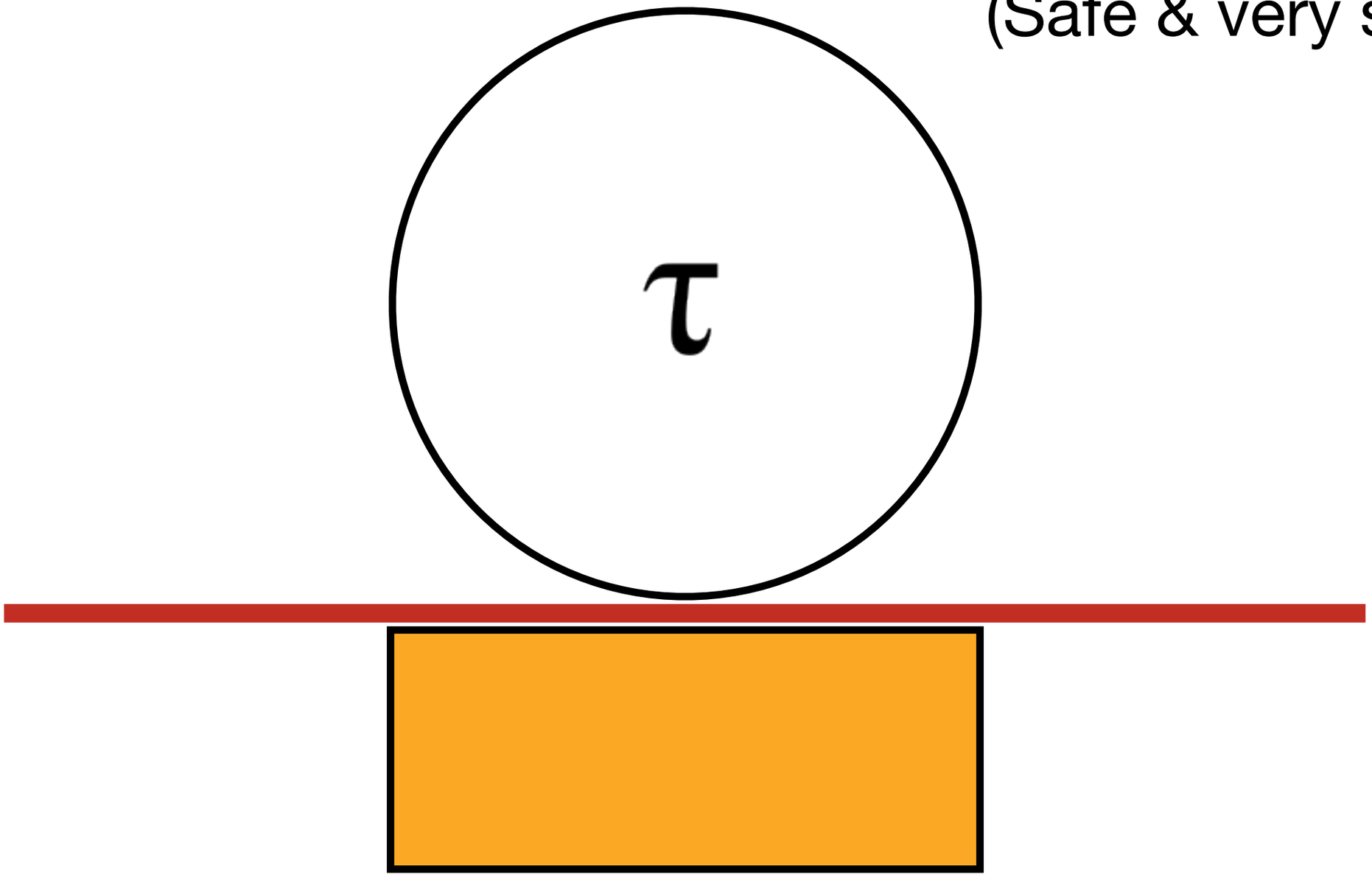
Option 1: Trusted Computing Base

(Bogus)



Option 2: Check Everything

(Safe & very slow)



Types + Soundness (revised)

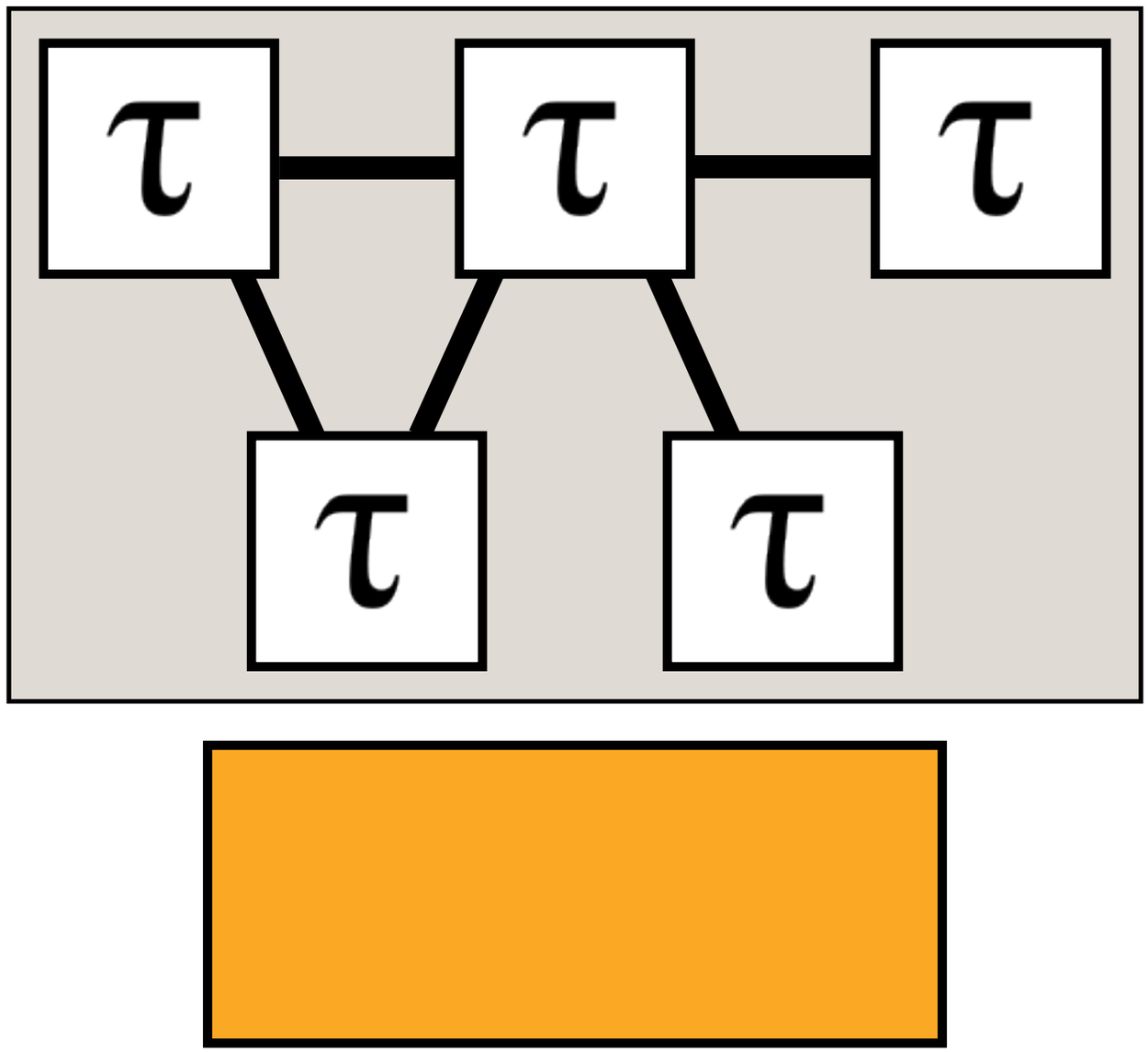
eliminate a class of errors from typed code

identify errors at typed-untyped boundaries

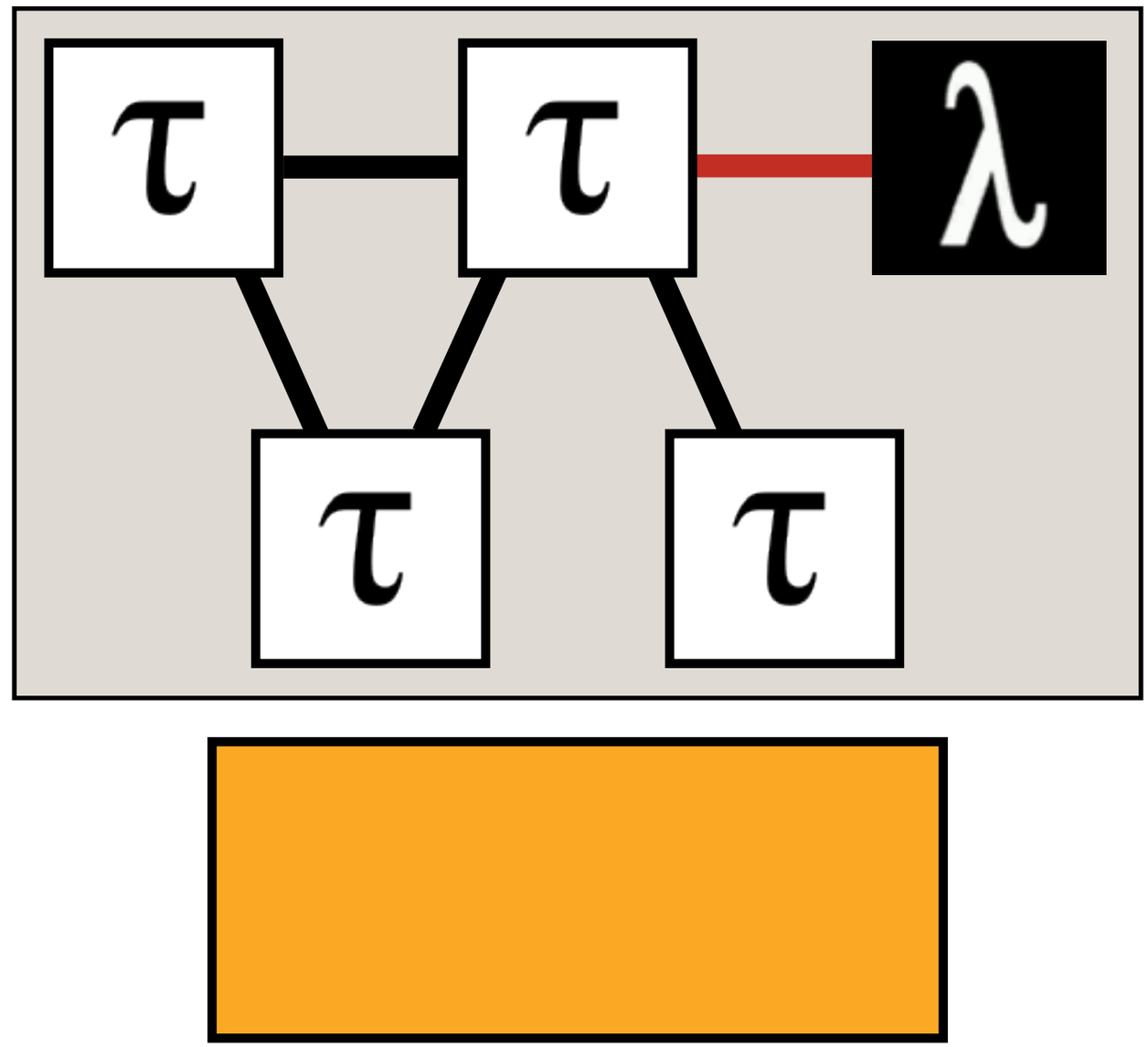
→ Option 1: Trusted Computing Base

Option 2: Check Everything

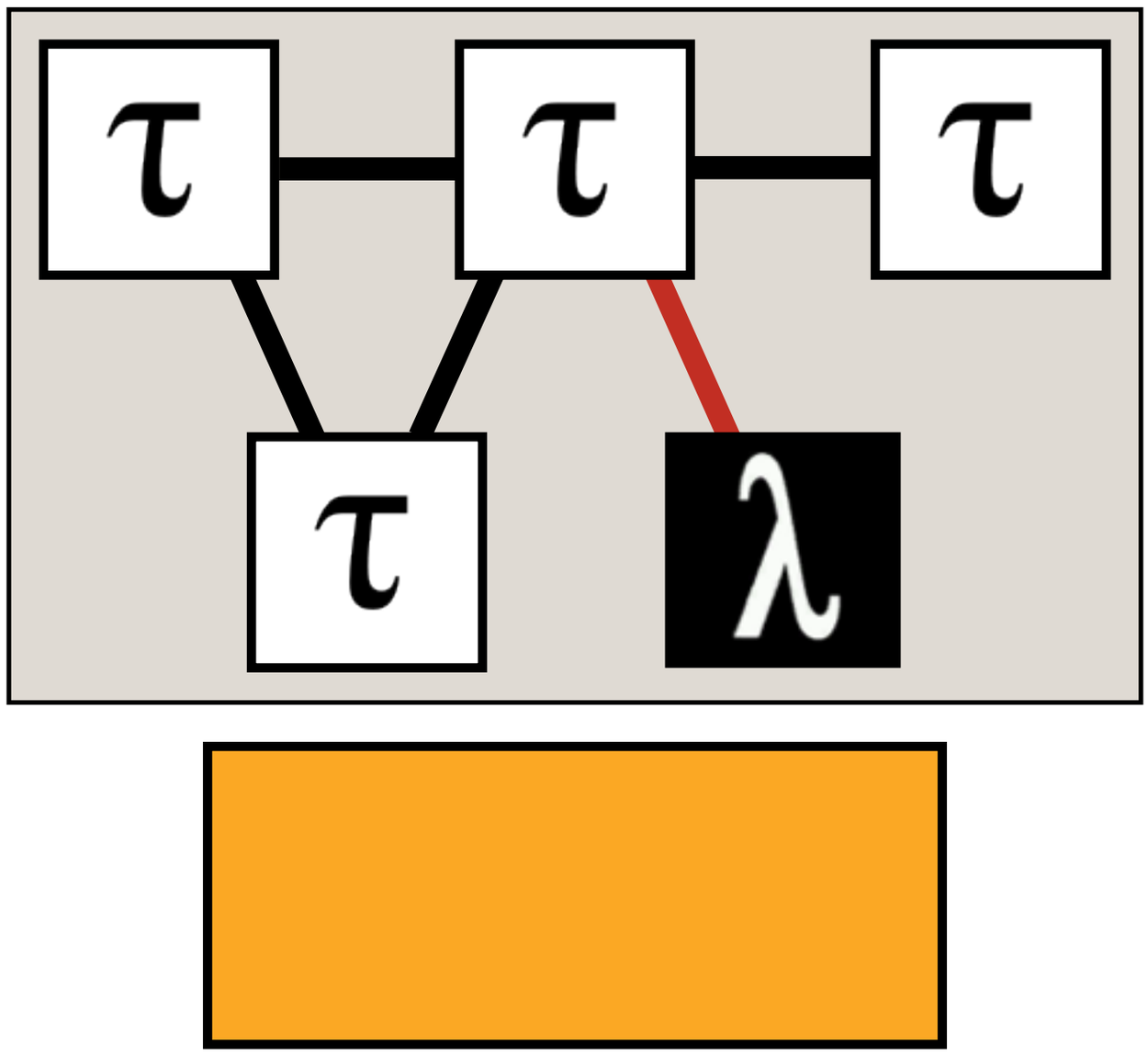
The Reality of (Macro) Gradual Typing



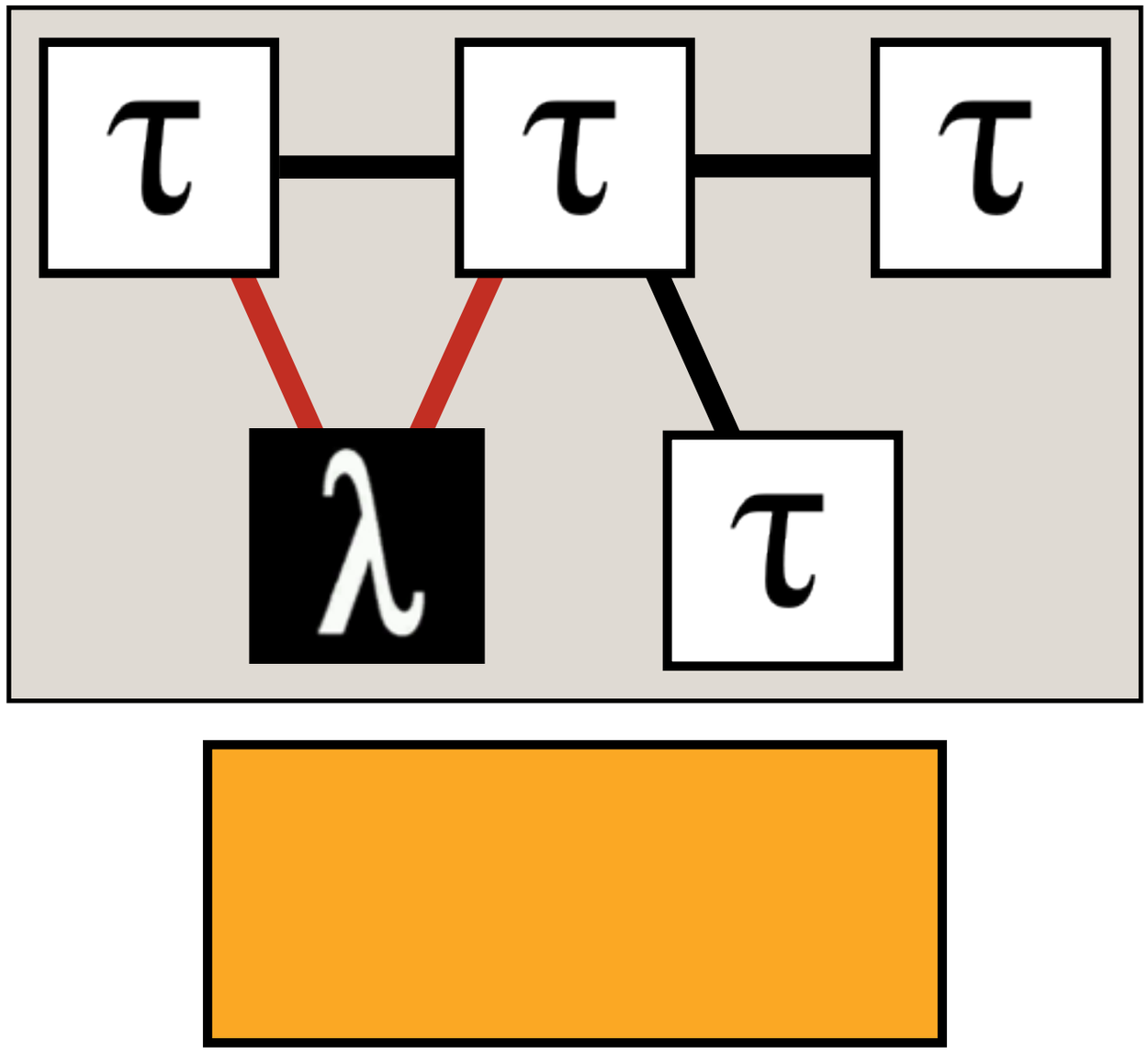
The Reality of (Macro) Gradual Typing



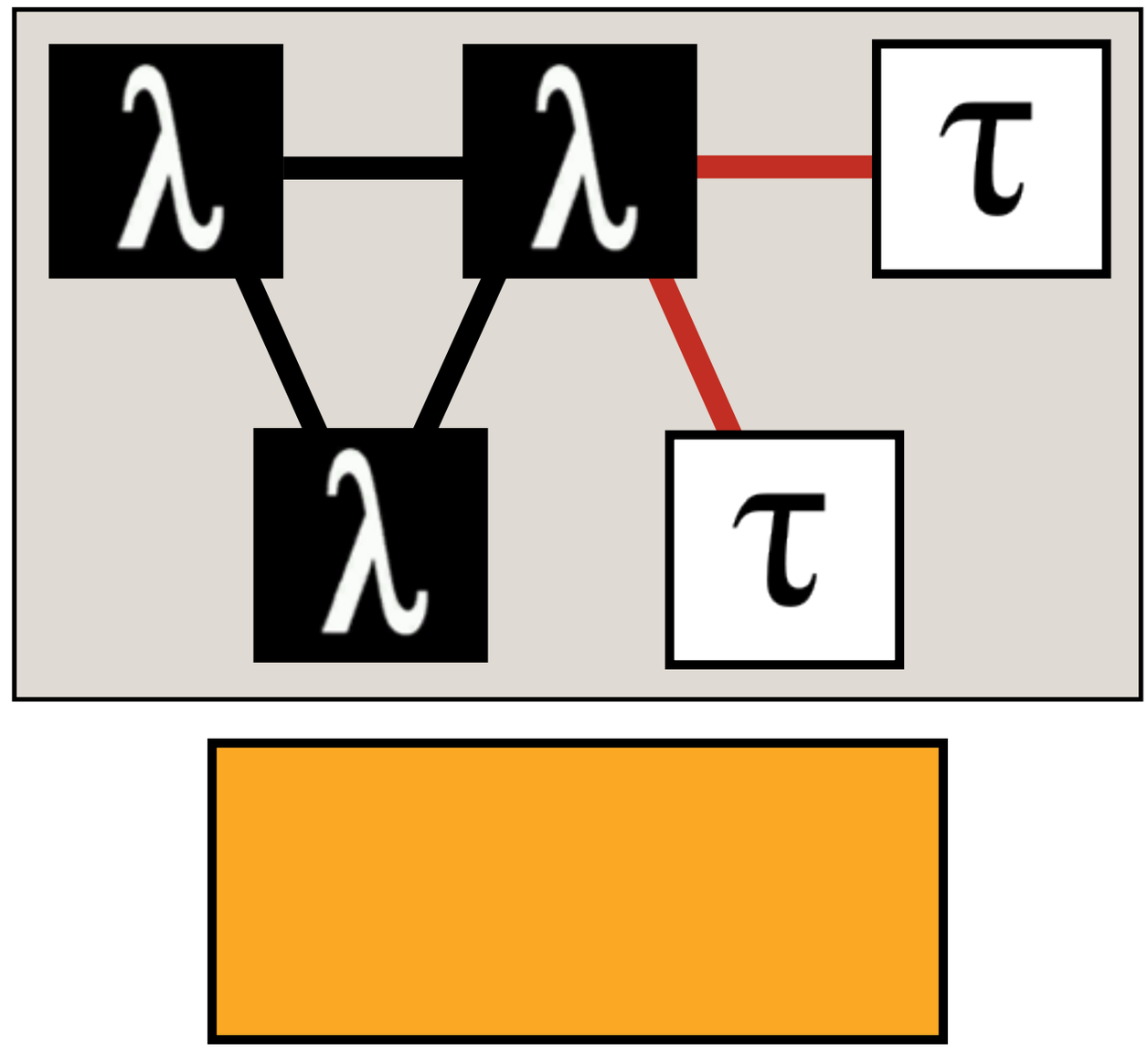
The Reality of (Macro) Gradual Typing



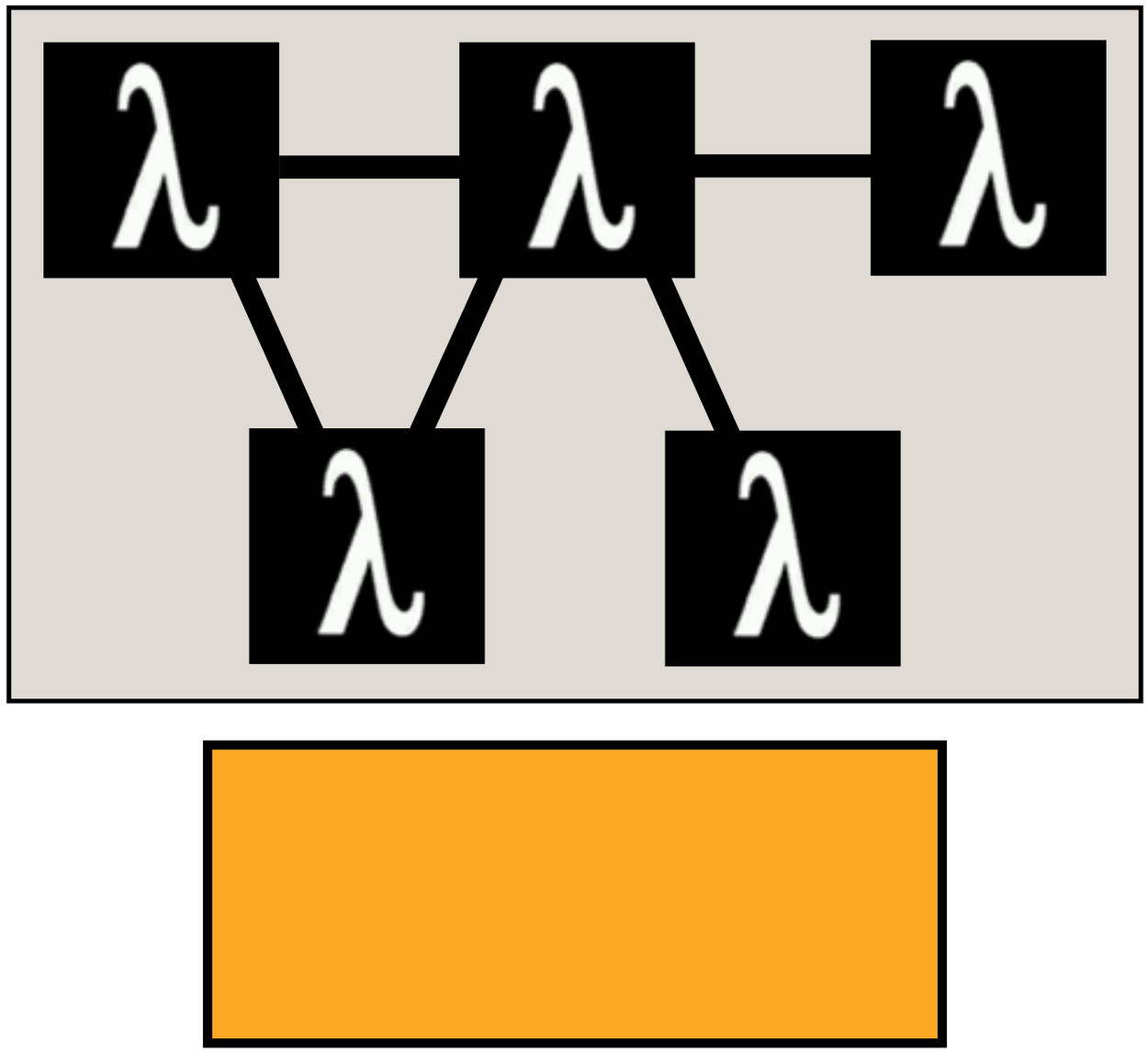
The Reality of (Macro) Gradual Typing



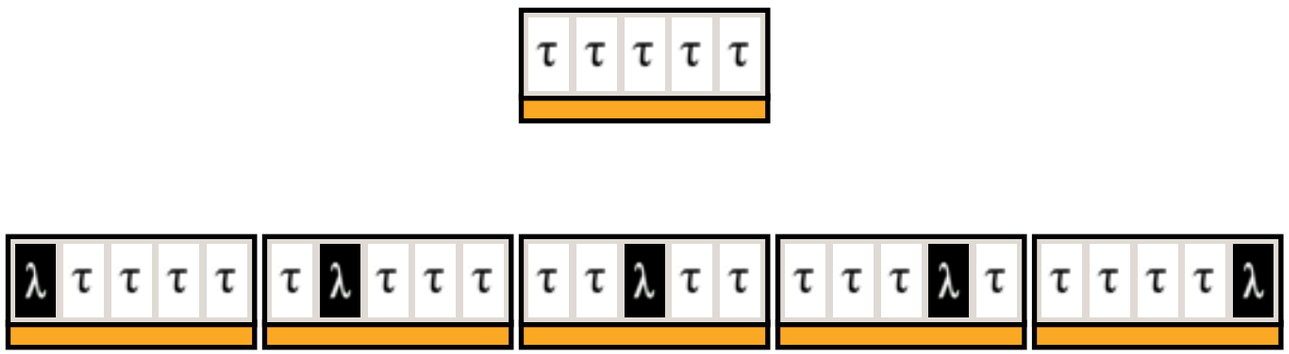
The Reality of (Macro) Gradual Typing



The Reality of (Macro) Gradual Typing



The Reality of (Macro) Gradual Typing



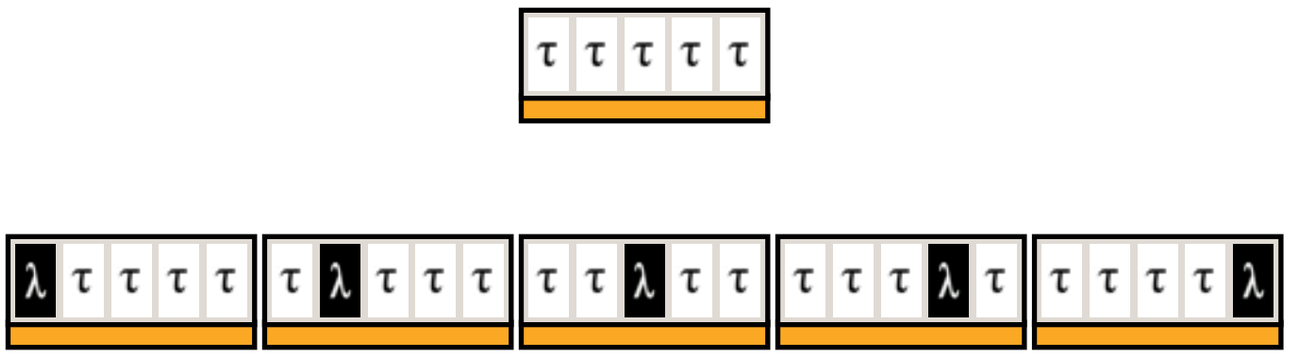
Recap

- Gradual type systems have "bad" performance

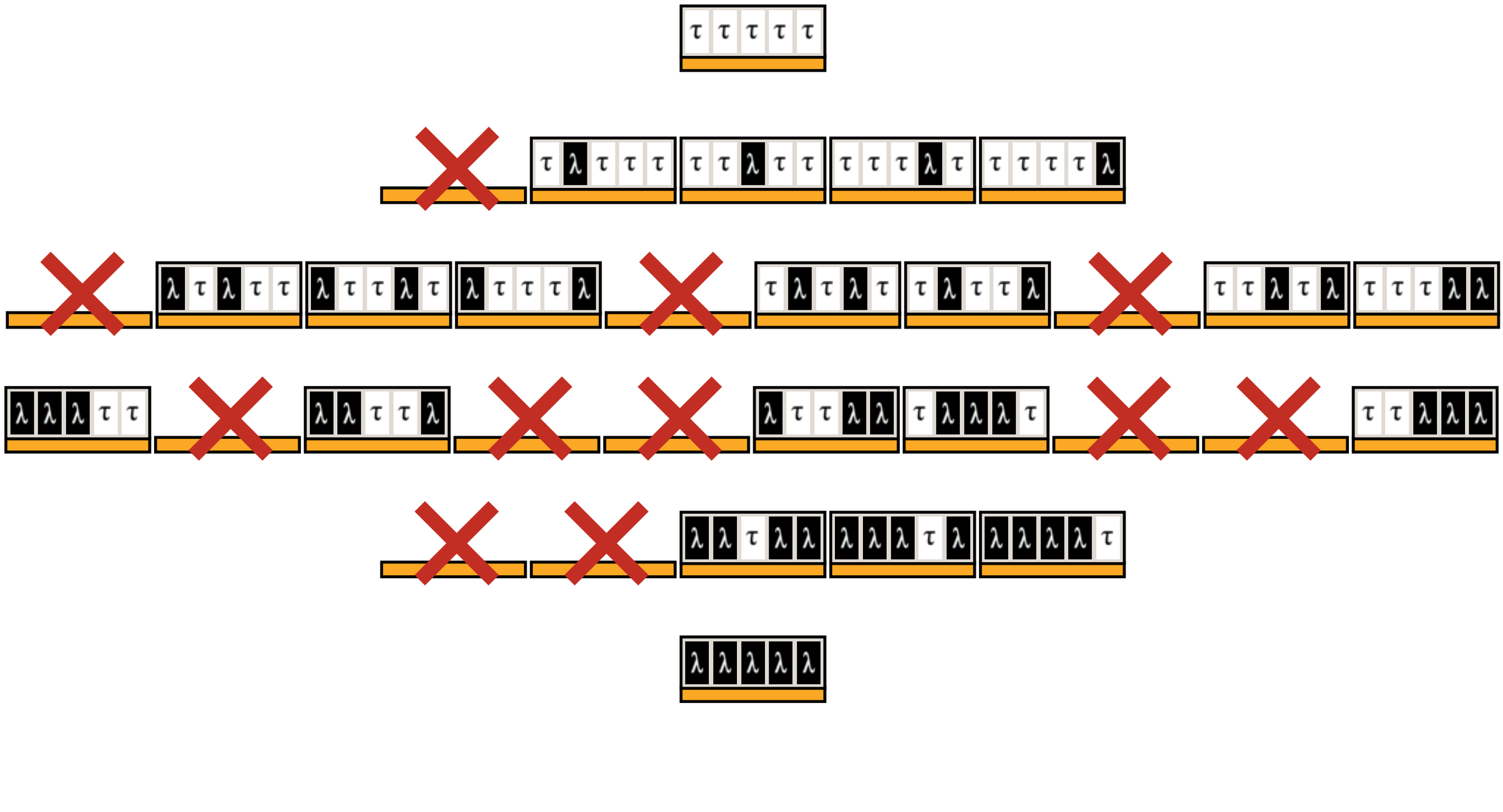
- Type soundness is imperative

- Issue: typed-untyped boundaries

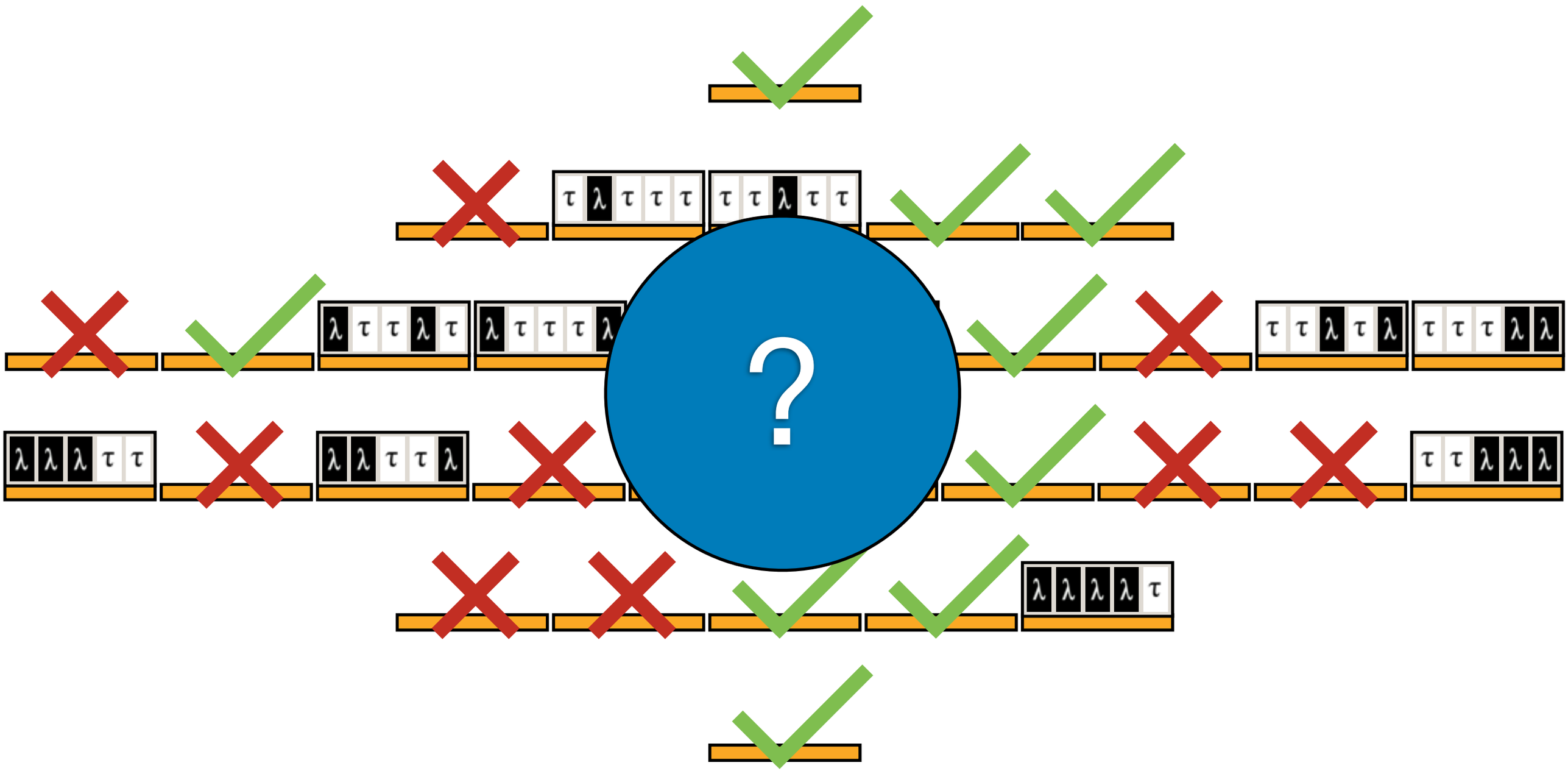
The Reality of (Macro) Gradual Typing



The Reality of (Macro) Gradual Typing



The Reality of (Macro) Gradual Typing



Proposal

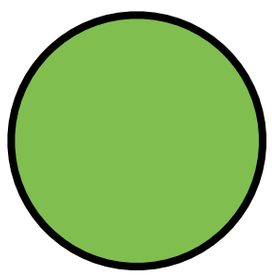
I. Define "boundary"

II. Run the **entire lattice**

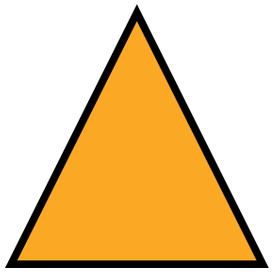
III. Define "performant"

IV. Report the % of performant variations

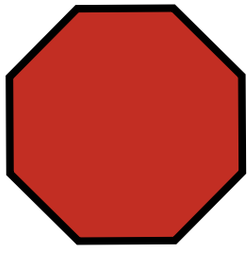
To be continued...



If you **agree**, then **use** our framework



If you **disagree**, then **propose** an alternative



Bottom line: no more "footnotes" on performance!

How to Scale?

- Hire a cluster

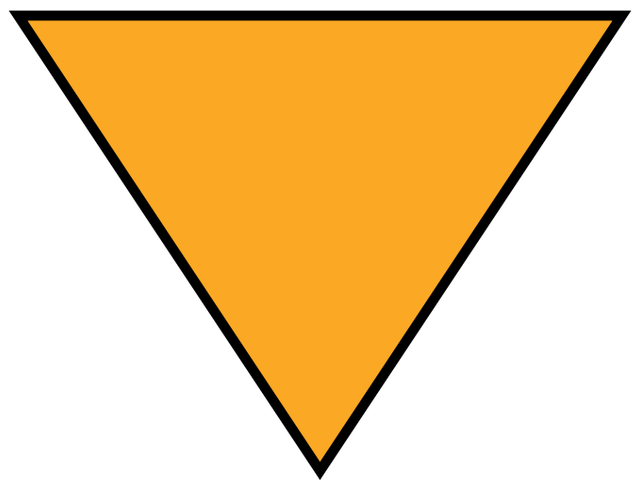
- Random sampling

- Offer fewer configurations

- Tools to identify / exclude configs.

Gradual Typing: Promise

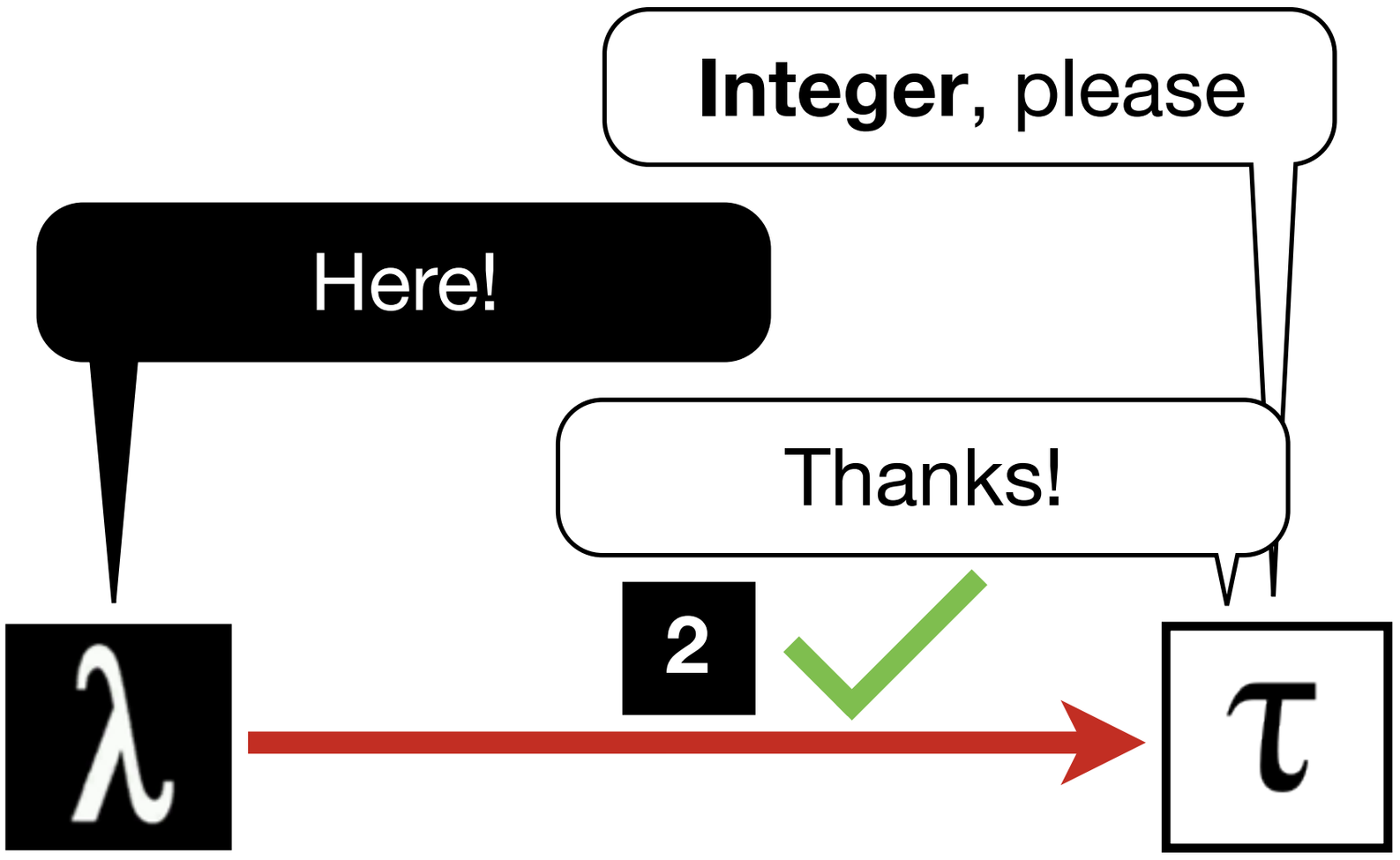
Expressive



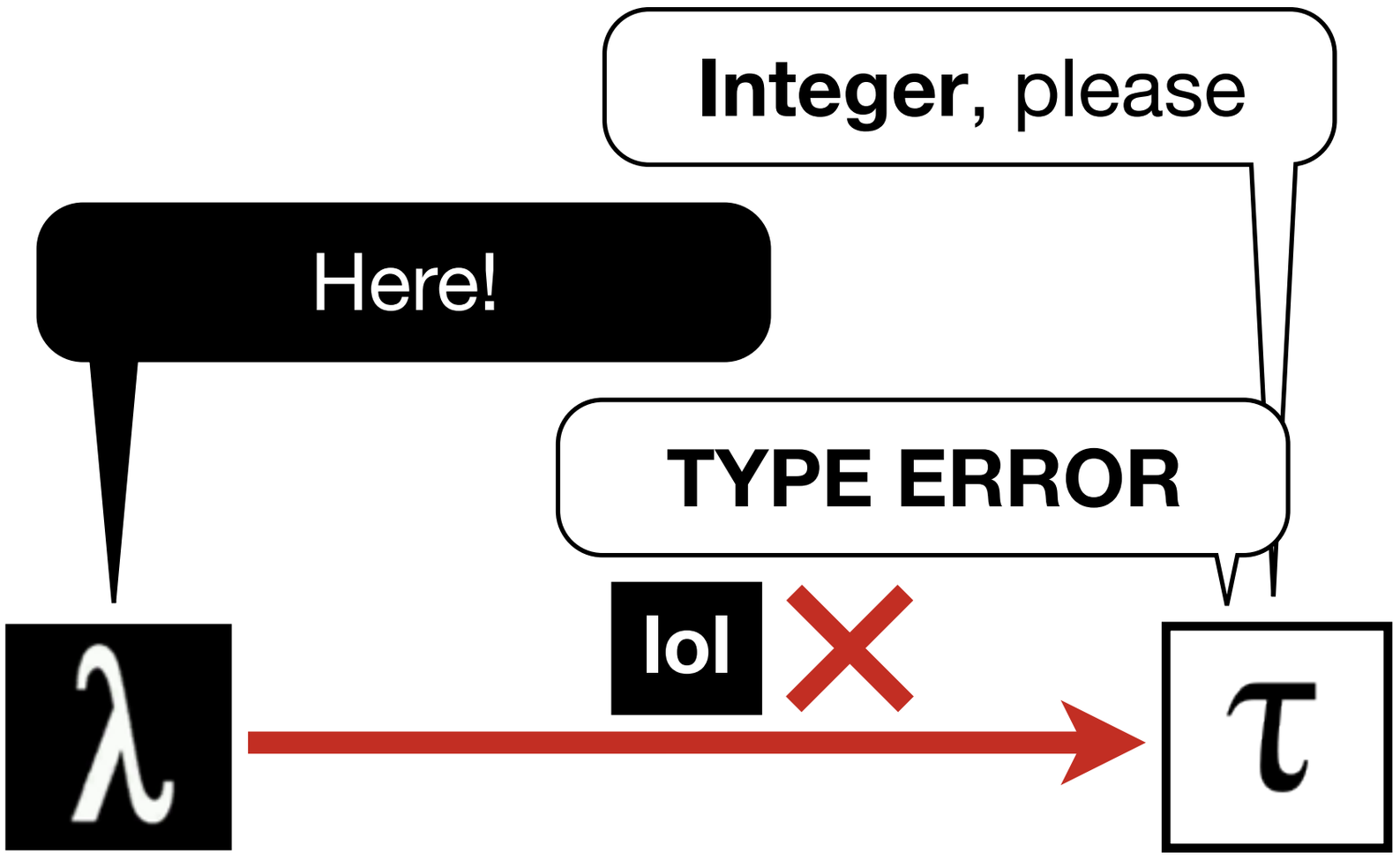
Sound

Performant

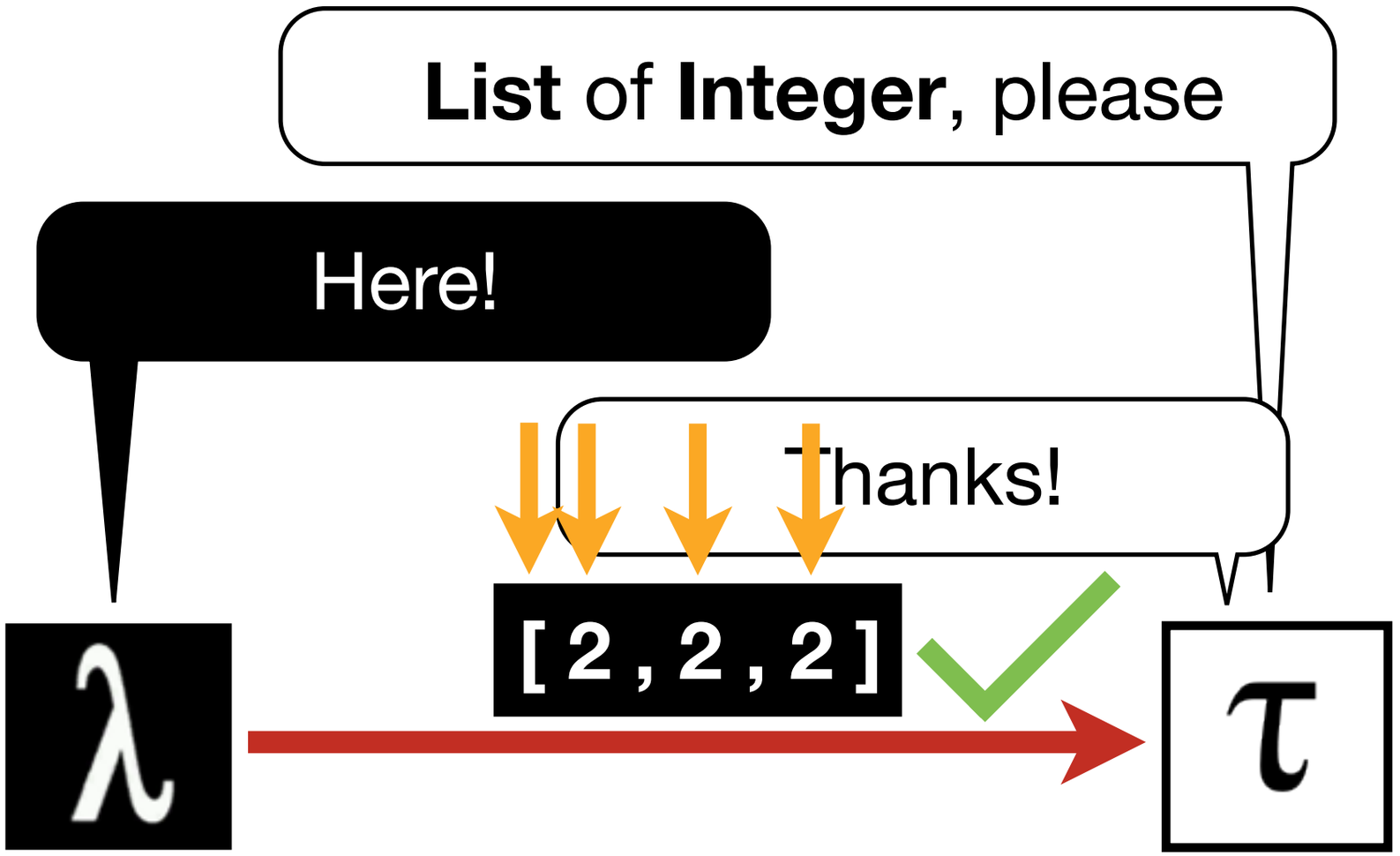
Boundaries



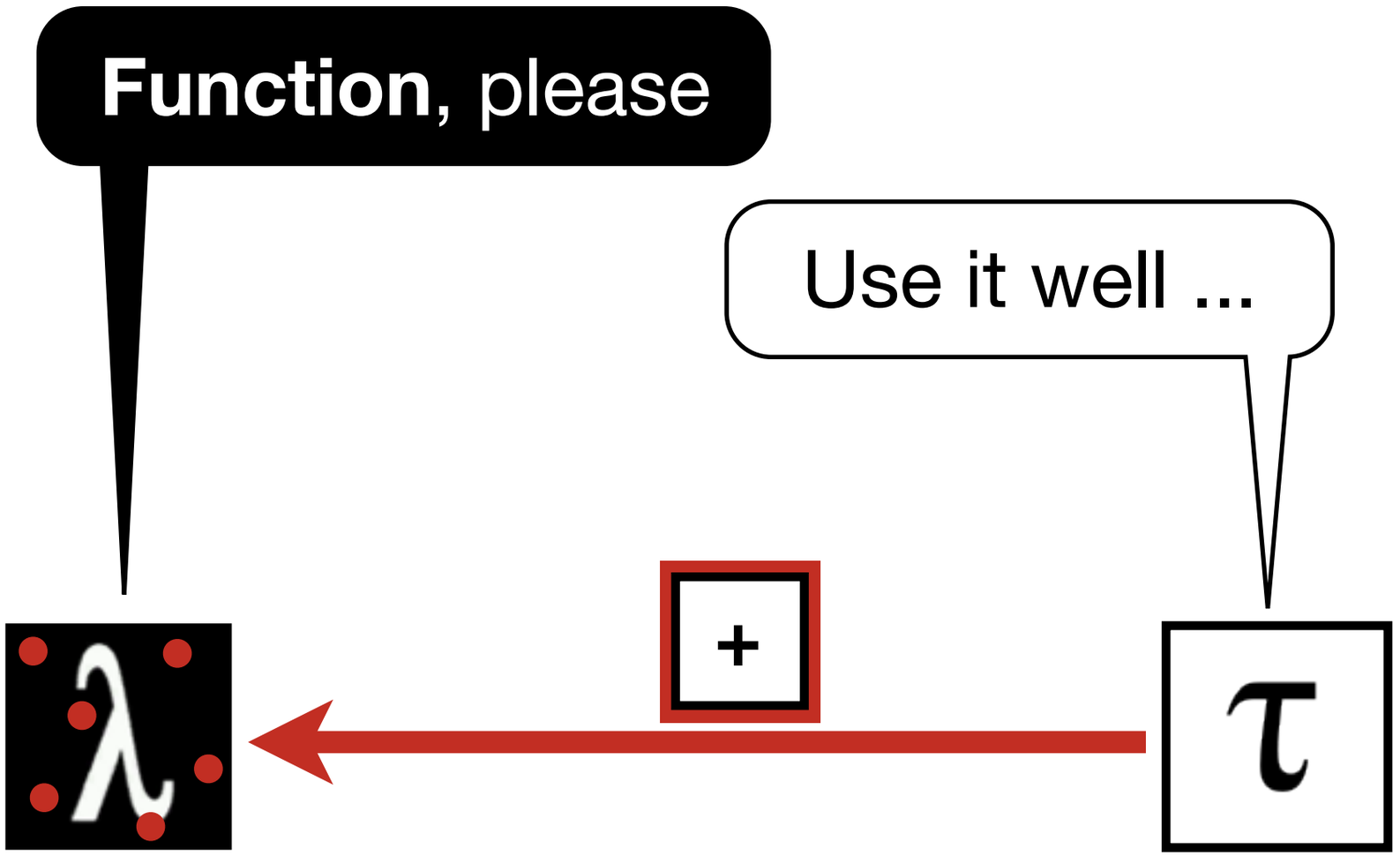
Boundaries



Boundaries



Boundaries



Reinheitsgebot

1 barrel of wine + 1 teaspoon of sewage

=

1 barrel of sewage