開心寫一個 Proof Assistant

西風 | Favonia

今日重點

依值型別為基礎的系統

Coq, Agda, Lean, ...

跟編譯器很像，但是...

關鍵是互動的過程

最終產生的機械碼（如果有的話）不是重點

瓶頸: 型別檢查

RedPRL團隊

Tulip S. Amalie
Carlo Angiuli
Evan Cavallo
Favonia
Matthew McQuaid
Reed Mullanix
Jon Sterling

RedPRL redtt cooltt algaett

再^{4}寫一個 Proof Assistant!

NbE* —— 檢查兩「項」是不是一樣

*Normalization by Evaluation

Refiner —— 相當於實作核心語法型態檢查的推理規則

*類型檢查 Part 1

Kernel —— 『受信任』的核心

Elaborator — 表層語法轉成核心語法

*類型檢查 Part 2

“Driver” —— 頂層定義、匯入其他檔案等等

Parser —— 字串轉成表層語法

NbE*

Refiner

Kernel

Elaborator

“Driver”

Parser

友人帳 Yuujinchou: 命名空間

バントーラ Bantorra: 函式庫管理

淺井 Asai: 編譯器診斷訊息

base/utility

algaeff: algebraic effects

bwd: backward lists 反向串列

type theory

無限 mugen: universe levels

カド kado: Cartesian cubical type theory 中的 cofibrations

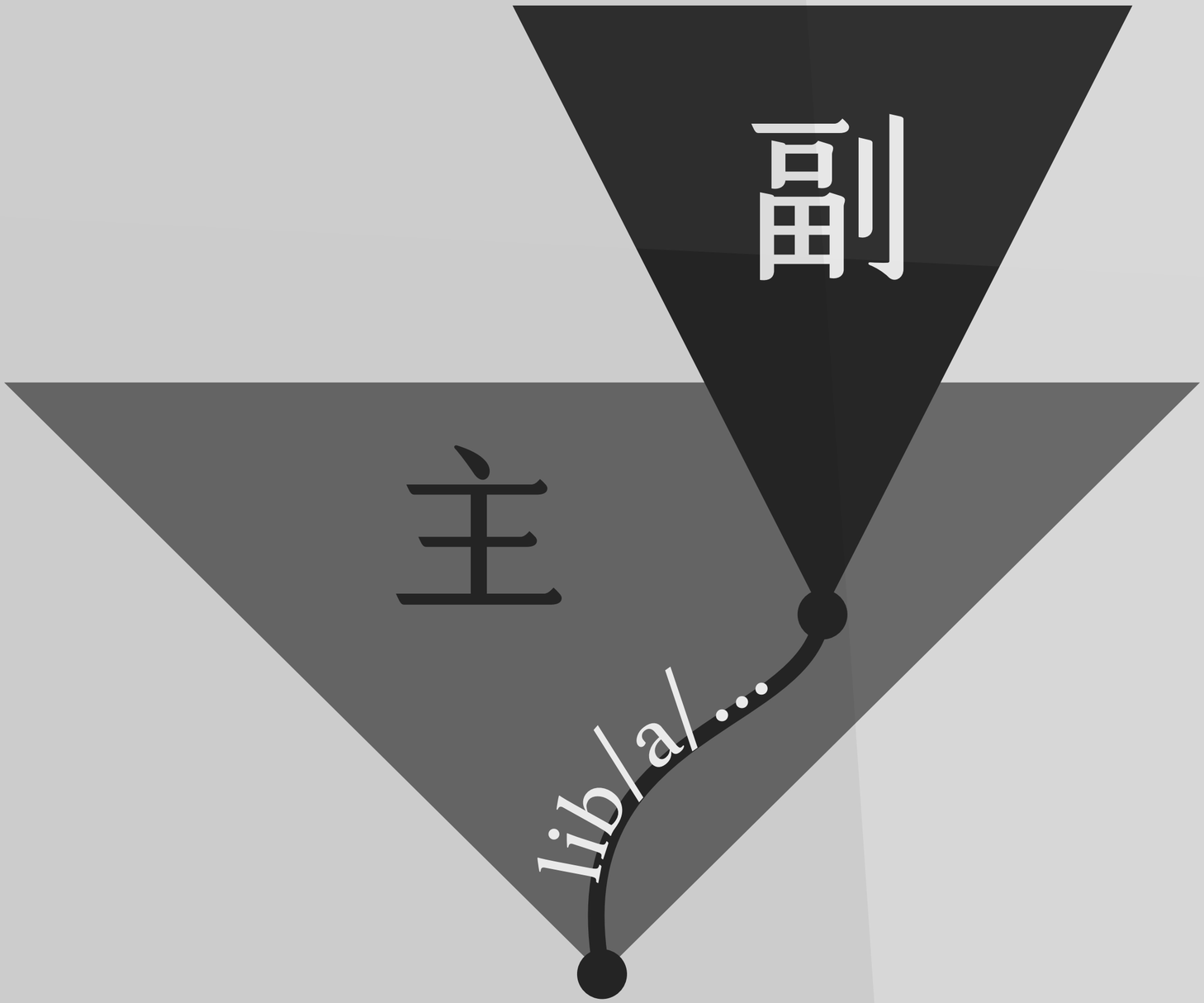
分進合擊

快速的為新的型別理論
寫好用的 Proof Assistant

Yuujinchou
友人帳
命名空間

Bantorra
バントーラ
函式庫管理

POSIX 風格遞迴解析
内建路由: local dir, git, index, ...
内建組合子: dispatch, rewrite, ...
執行緒安全
Algebraic effects


副
主
lib/a/...

Asai
淺井

編譯器診斷訊息

完美支援 Unicode: 永遠不要問「第幾欄」

多後端: LSP*, Terminal, ...

*currently broken

堆疊回溯

多段標記

Algebraic effects

Asai

淺井 API 設計

一個診斷訊息要包括什麼資訊？

1. 程式應該立即結束嗎？
2. 訊息分類：如何呈現給終端使用者（錯誤或警告？）
3. 一個容易 Google 的錯誤代碼
4. 堆疊回溯+多段標記

以上全包，然後 API 還要清爽乾淨

Mugen
無限
上禮拜講過囉

負層: 每個 universe 都有一個更小的 universe

有理層: 任兩層之中有無限多層可以用

碎型層: 可以把整個階層架構鑲嵌在任兩層之中

Kado
カド
cofibrations

高度最佳化: 目前用於 cooltt
同時用來實作「定義展開控制」

欲罷不能

Records
Pretty Printing
Inductive Types
Meta/Staged Programming
…

Demo