Building Blocks for Reasoning on the BEAM

IWIL 2026, Spetses

Johannes Schuster and Christoph Benzmüller
AI Systems Engineering, University of Bamberg
25 October 2026

Introduction

OCaml	Satallax, Lash, LEO-II, Zipperposition
C/C++	E, Vampire, cvc5
Scala	Leo-III
the BEAM	-

A prover on the BEAM starts with

terms	THF parsing	unification	term ordering	external provers

Erlang, Elixir, Livebook

The BEAM is the virtual machine of Erlang/OTP.



runtime and libraries:
processes, supervision,
distribution



a language on the BEAM;
Shot and the packages



notebooks for Elixir;
the thesis and the
demonstrations

The BEAM

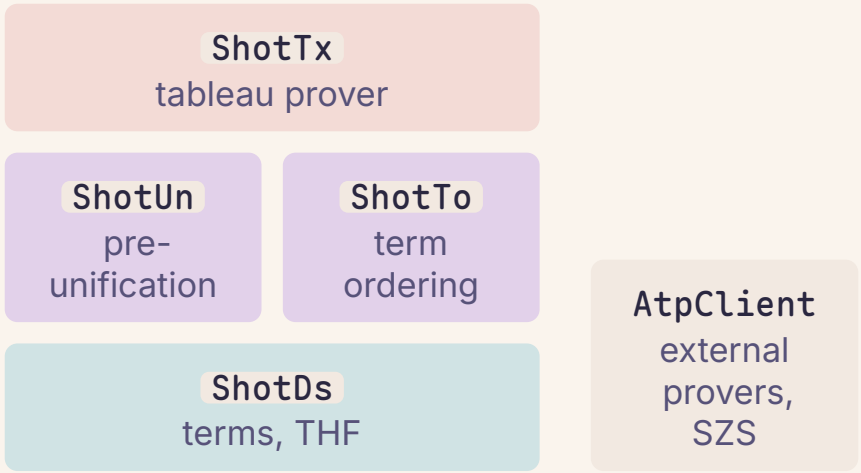
offers	lightweight processes, preemptively scheduled, each with its own heap; actor model; supervision; message passing
costs	a constant factor for serial work

Hypothesis: a proof search decomposed into independent concurrent work recovers the constant factor.

Shot

logic	Classical HOL; TPTP THF
calculus	free-variable tableau with unification; *binding-free* rules
status	ShotTx 0.1.2 on Hex; usable in Elixir or CLI

Infrastructure



`ShotDs`	hash-consed $\beta\eta$-long terms, de Bruijn indices, parsing
`ShotUn`	depth-bounded Huet pre-unification, pattern path
`ShotTo`	NCPO-LNF as a decidable predicate
`AtpClient`	one API, SZS results

On Hex: `shot_ds` 1.3, `shot_un` 0.2, `shot_to` 0.2, `shot_tx` 0.1, `atp_client` 0.8.

Calculus

$$\frac{\varphi \wedge \psi}{\varphi, \psi}$$

α

$$\frac{\varphi \vee \psi}{\varphi \mid \psi}$$

β

$$\frac{\forall x_\tau.\varphi}{\varphi[X_\tau]}$$

γ, X fresh, repeatable

$$\frac{\exists x_\tau.\varphi}{\varphi[f\bar{X}]}$$

δ, f fresh over the free variables

$$\frac{\forall x_\tau.\varphi}{\varphi[\lambda \bar{y}.c(\bar{H}\bar{y})]}$$

primitive substitution, c logical, $\bar{H}$ fresh

$$\frac{\forall x_\tau.\varphi}{\varphi[t_1], \ldots, \varphi[t_k]}$$

γ over a finite domain: τ built from o, $\tau = \{t_1, \ldots, t_k\}$

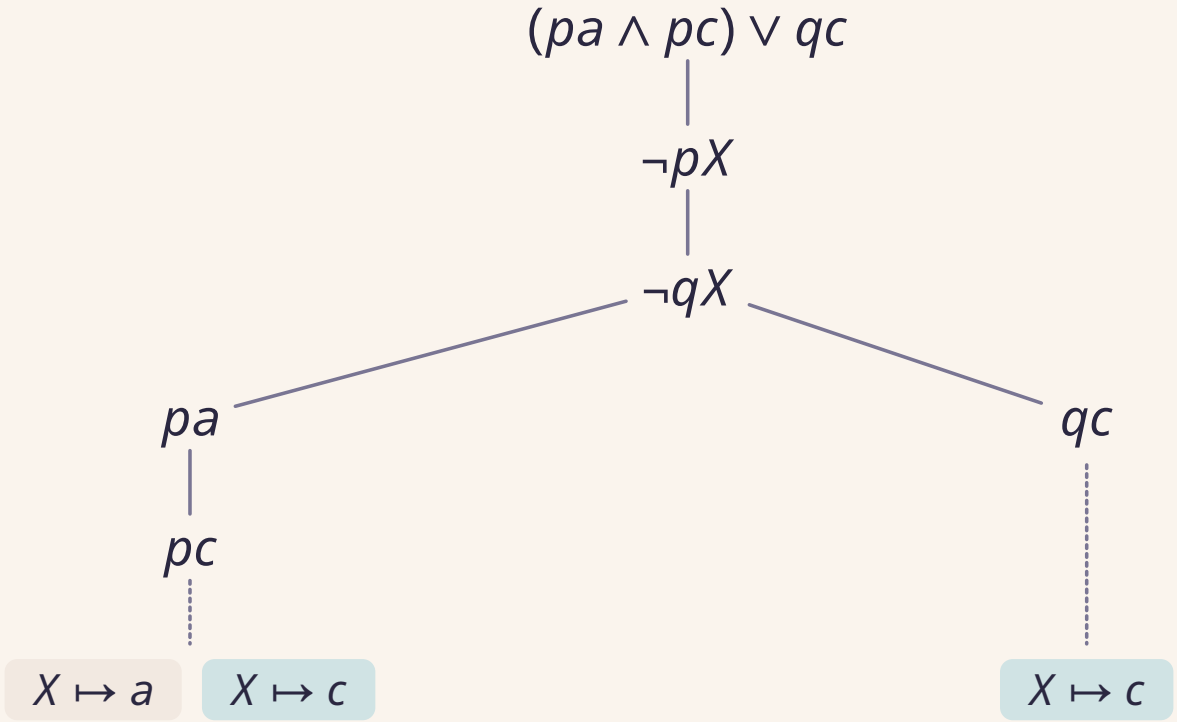
$$\frac{p\bar{s}, \neg p\bar{t}}{\times}$$

closure, $\bar{s}$ and $\bar{t}$ identical or unifiable

Rules ordered by hand-written cost function

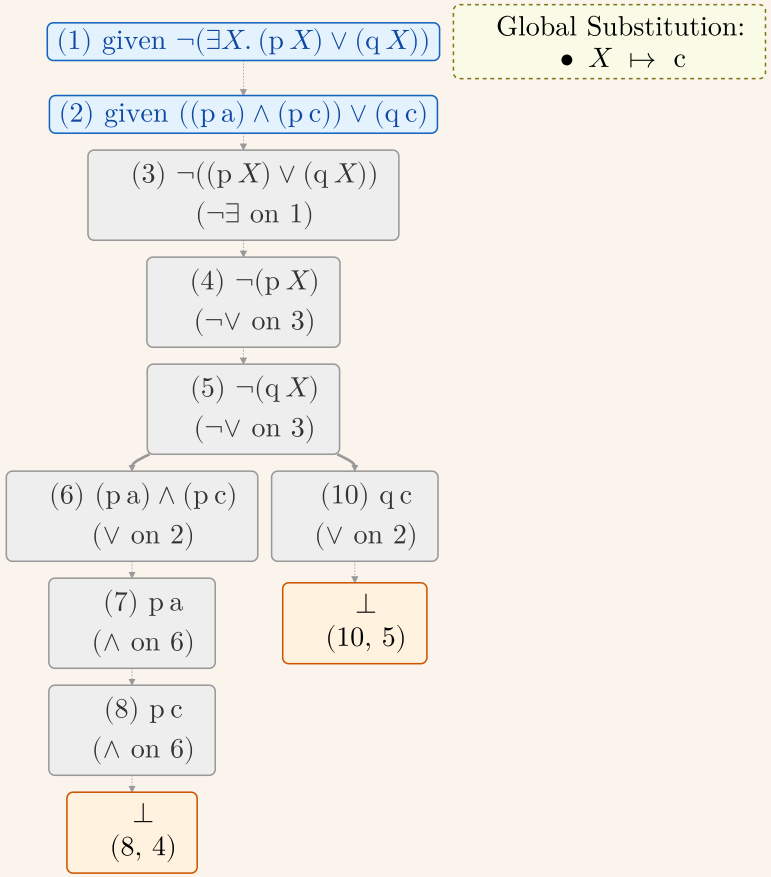
Worked example

$((pa \wedge pc) \vee qc) \supset \exists X.pX \vee qX$



The example in Shot

branches	each records its pair as evidence: (8, 4) and (10, 5)
closure	one search over both: $X \mapsto c$
output	`ShotTx.Proof.to_mermaid/2`



Binding-free rules

No branch rule applies a substitution.

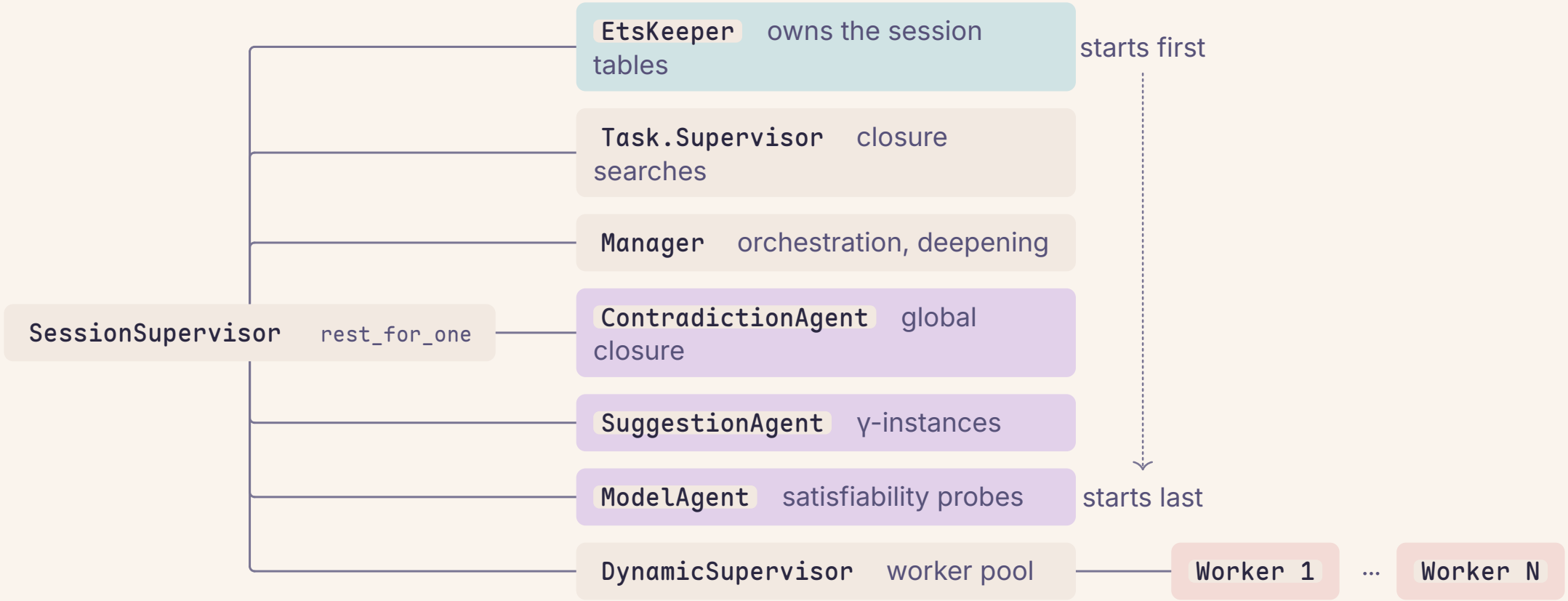
evidence	disagreement pairs are recorded, not solutions
demodulation	ground only, by identifier equality
flex–flex pairs	deferred
comparisons	substitution-stable only

Agentic processing

No substitution: a branch is an immutable value in a shared table.

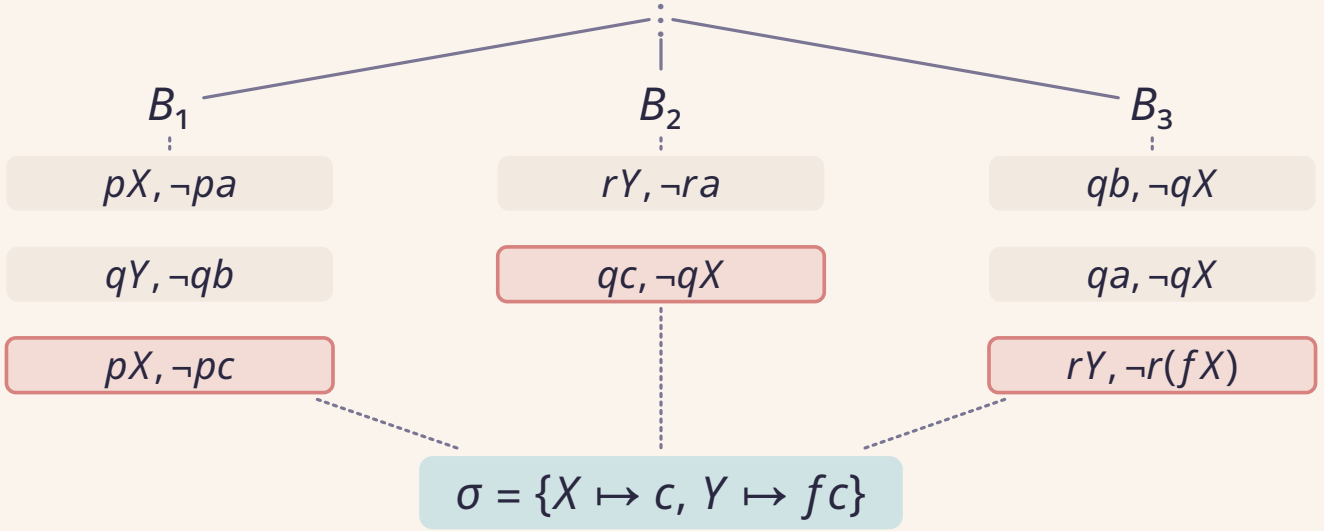
worker	contradiction agent	suggestion agent	model agent
expands	reads evidence	proposes γ-instances	probes saturated branches

Supervision tree



Global closure

state	option lists per open branch, held by the contradiction agent
closure	one option per branch, simultaneously unifiable
search	lazy cartesian product, first solution or deadline



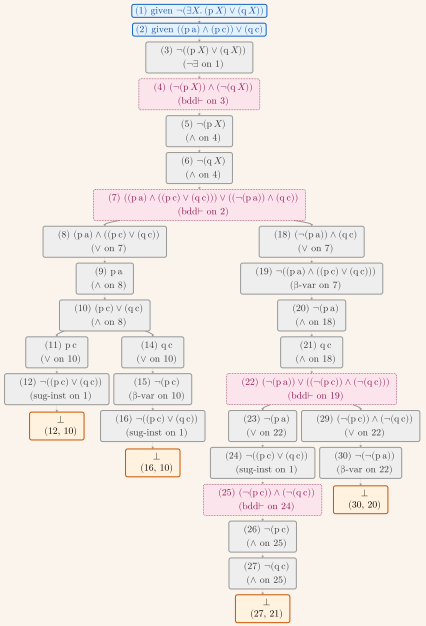
Demonstration

call `ShotTx.Prover.prove/2` on the example as a THF problem

output `ShotTx.Proof.to_mermaid/2`

```
{:thm, proof} =
  ShotTx.Prover.prove(problem, timeout: 5_000)

proof
|> ShotTx.Proof.to_mermaid()
|> Kino.Mermaid.new()
```

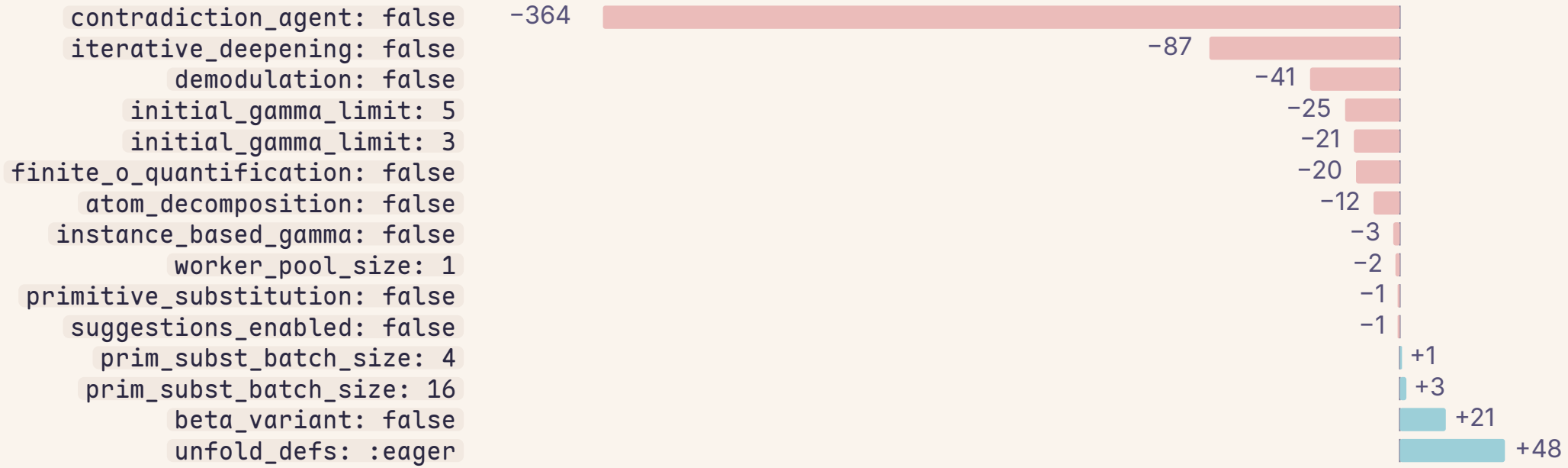


The example under the default configuration: 31 steps, closed by ground instances from the suggestion agent.

Evaluation

problems	TPTP TH0 and TH1, 5 109
reach the prover	4 183 (628 parse timeouts, 221 parse errors, 77 without conjecture)
time limit	2 000 ms
scored	refutations and saturations
run-to-run	std = 5.3 problems between two single runs (structured set, three repeats)

Ablation



Refutations relative to the default (682).

Refutations+Saturations: 743 default, 905 over union of configurations.

Structured set (Benzmüller & Brown, 2005)

prover	proved of 130
Vampire	121
Leo-III	120
Satallax	118
E	116
Zipperposition	112
Shot	79

model class	in every run	of
without fun. ext. principles	57	70
with fun. ext. principles	5	35
other	14	25

timeouts: Shot 5 s, reference provers 10 s via SystemOnTPTP.

Worker pool

8 vs 1 worker	pool	single	ratio
steps	5 414 451	1 654 858	3.27
branches closed locally	121 798	38 853	3.13
global closure searches	2 469	2 115	1.17
global closure searches per 1 000 steps	0.46	1.28	0.36

Refutations: 680 single, 682 pool.

Contradiction agent

contradiction agent, probed (16 schedulers)	pool	single
time in search	46.5 %	51.6 %
branches per search	21.41	10.37
candidates per search	996.84	161.05
searches abandoned	17.8 %	10.8 %
triggers suppressed per search	595	81

More inference, more evidence, longer searches, more discarded evidence.

The BEAM as a Platform

layer		verdict
orchestration	✓	pays off
interactive work	✓	pays off
search, local half	✓	scales with the pool
search, global half	✗	unification problem too hard for single instance

The verdict-deciding step must scale with the work feeding it.

Integration

`AtpClient`	unified API across several backends; truth grounding
`AtpMcp`	backends as tools e.g., for coding agents
`KinoAtpClient`	Livebook, teaching

Current and Future Work

`tptp` 0.3.0	parsing separated from elaboration, BNF-generated grammar, CST with spans
`HolTk`	a common base layer on `tptp` for Shot and further tools; stateless, serialisable data structures
Shot 0.1.2	conservative countermodels, TPTP-compliant CLI; the last release of this design
Shot redesign	destructive substitution, OR-parallel branches, extended signature

Conclusion

binding-free rules	branches become shareable data; closure moves out of the branch
hypothesis	not confirmed for proof search in this implementation; the BEAM pays off for orchestration and interaction
search	the bottleneck is the single arbitration point; it must scale with the work feeding it

Five packages on Hex. Thesis as Livebooks.

`{atp-client,shot-ds,shot-un,shot-to,shot-tx}.hexdocs.pm`

`github.com/jcschuster/shot-thesis`