Towards the Compression of First-Order Resolution Proofs by Lowering Unit Clauses

J. Gorzny[1] B. Woltzenlogel Paleo[2]

[1]University of Victoria

[2]Vienna University of Technology

6 August 2015

Our Goal

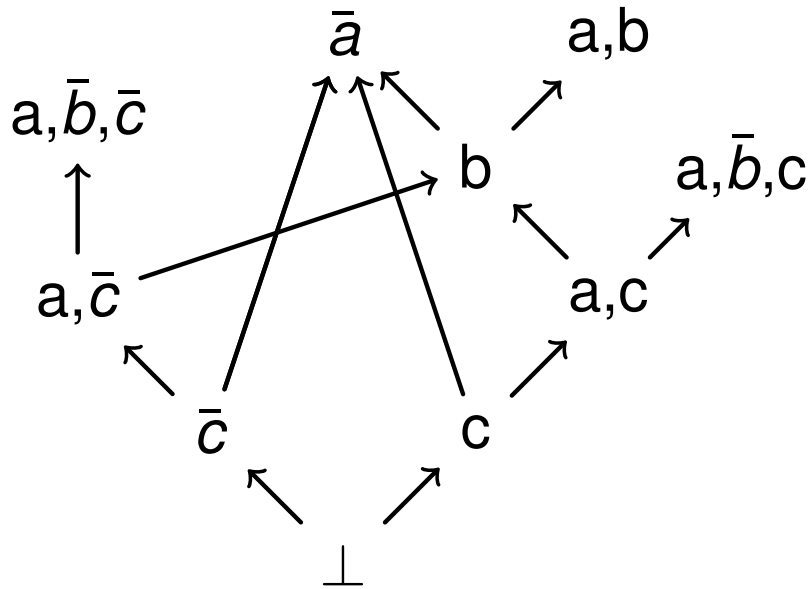
Lifting propositional proof compression algorithms to first-order logic.

This work: LowerUnits

Proof Compression Motivation

- The best, most efficient provers, do not generate the best, least redundant proofs.
- Many compression algorithms for propositional proofs; few for first-order proofs.

A Propositional Proof



LowerUnits

Definition (Unit)

A unit clause is a subproof with a conclusion clause (final clause) having exactly 1 literal

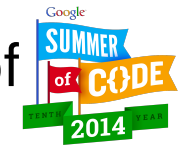
Theorem

A unit clause can always be lowered

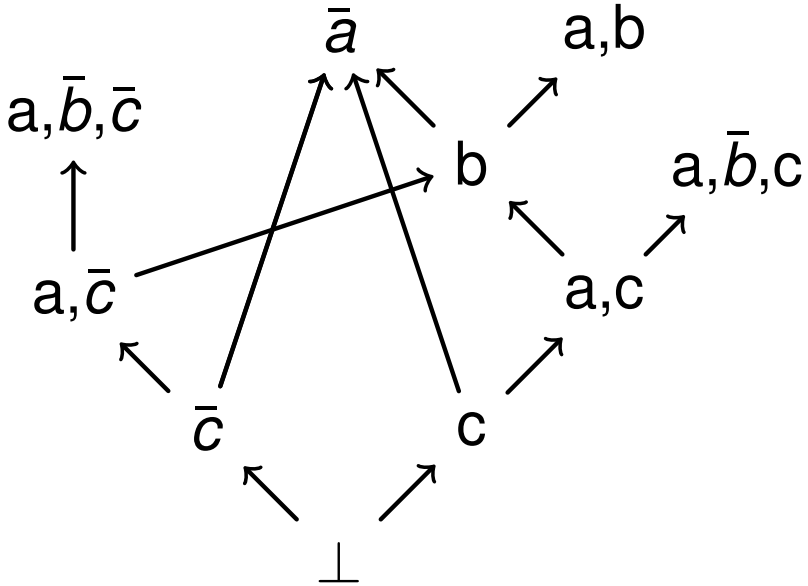
Compression is achieved by delaying resolution with unit clause subproofs.

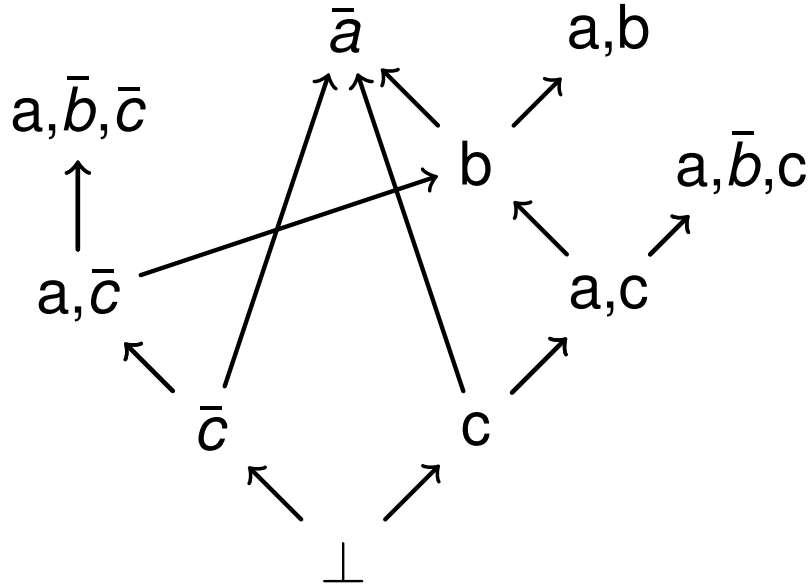
Two Traversals

- $\uparrow$ Collect units with more than one resolvent
- $\downarrow$ Delete units and reintroduce them at the bottom of the proof

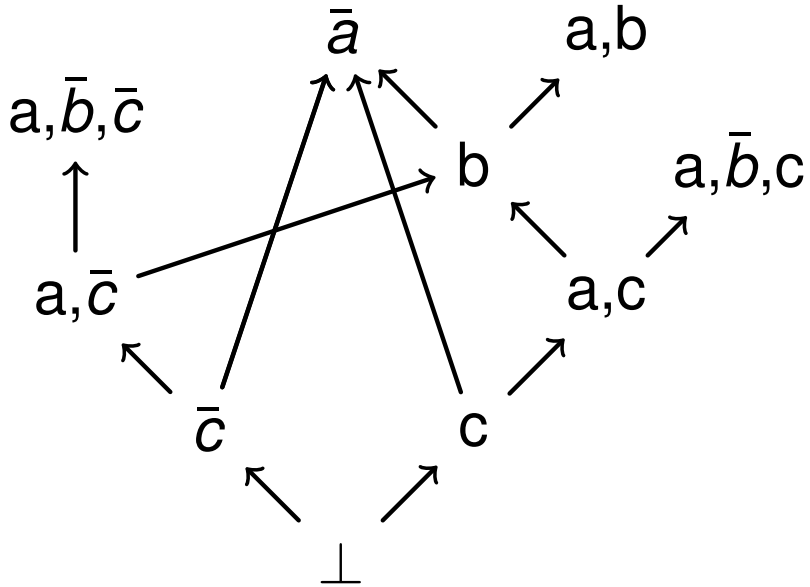


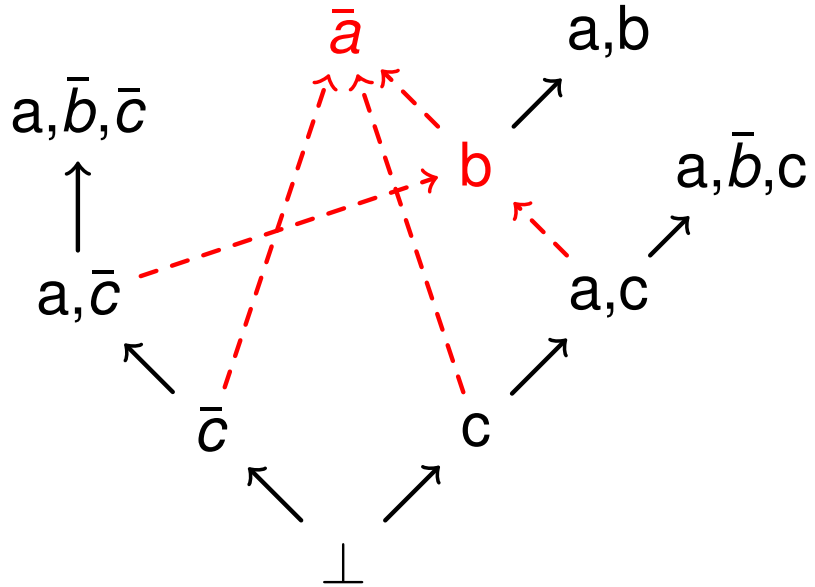
Propositional Example



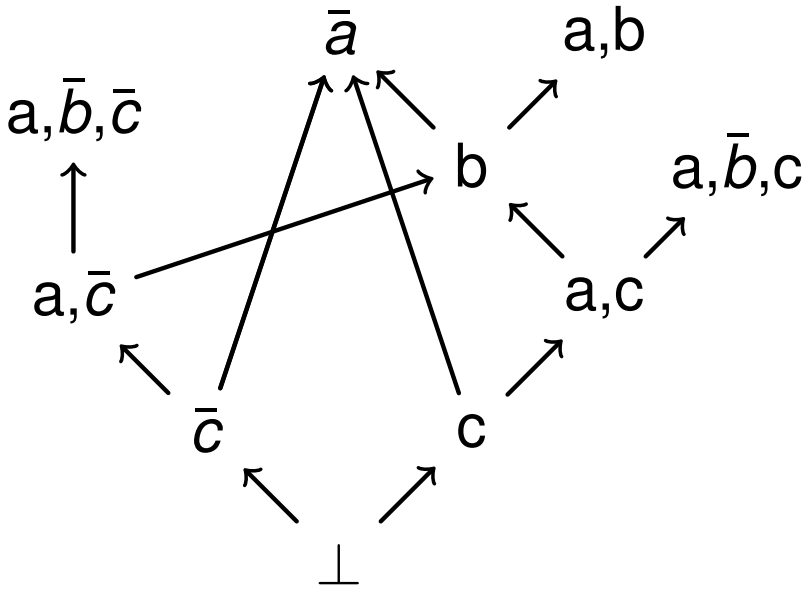


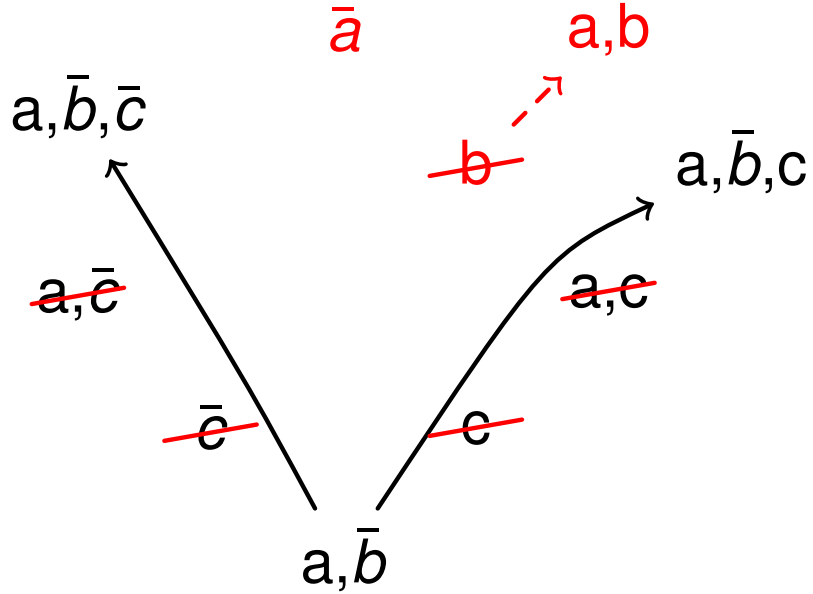
Propositional Example



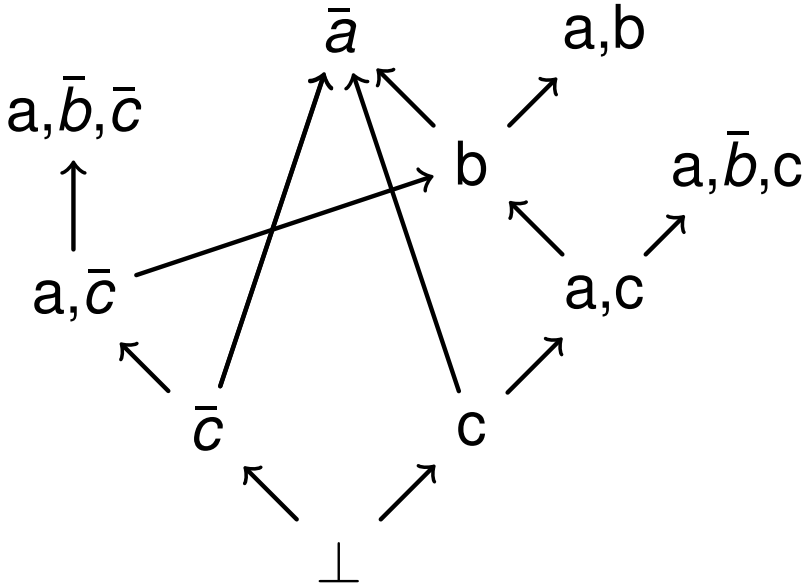


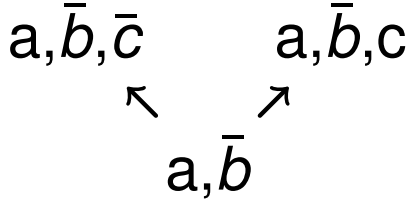
Propositional Example



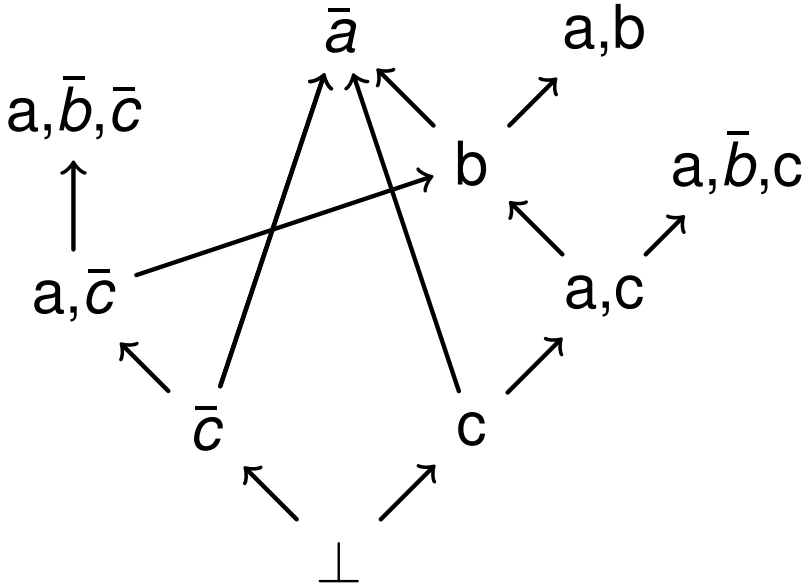


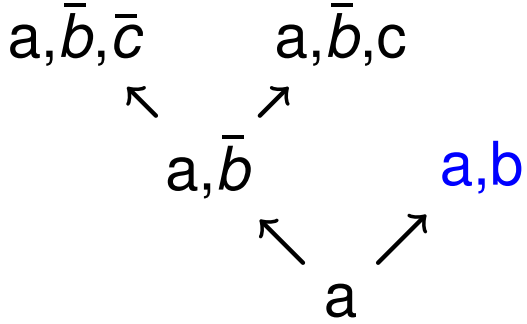
Propositional Example



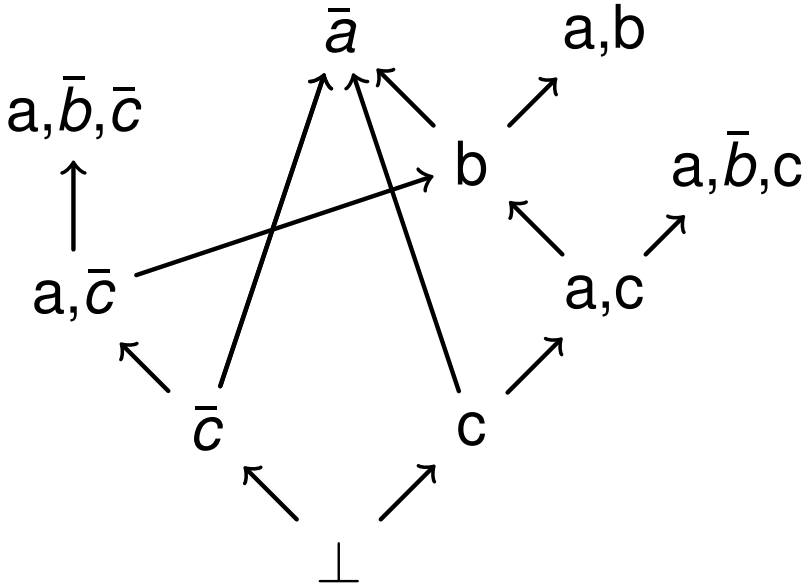


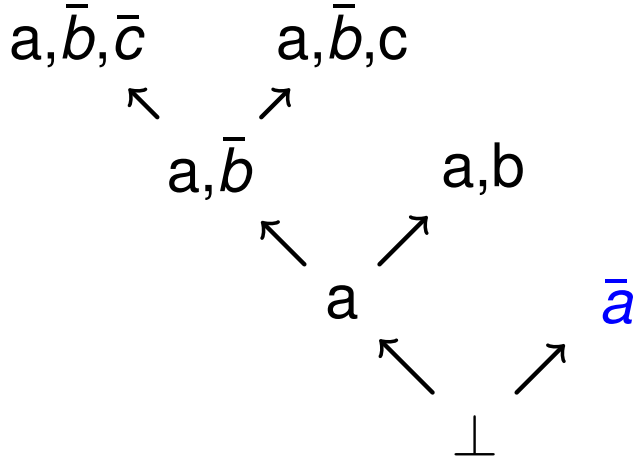
Propositional Example



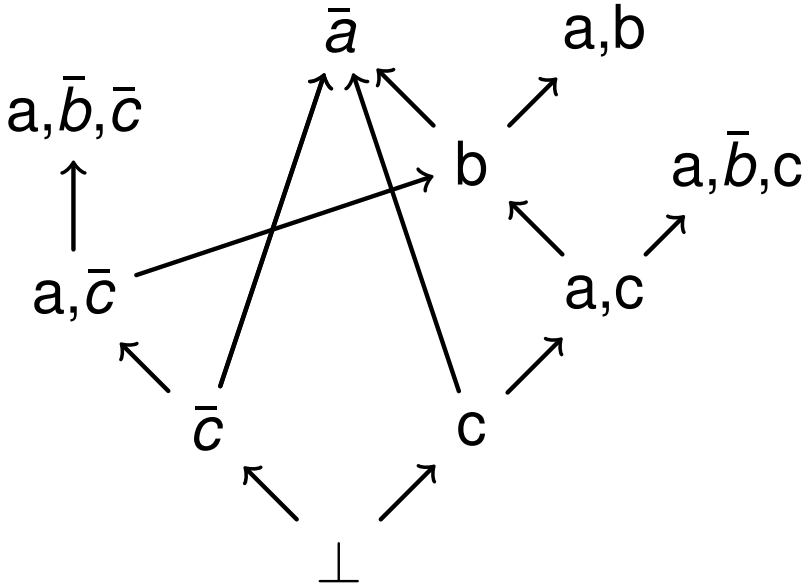


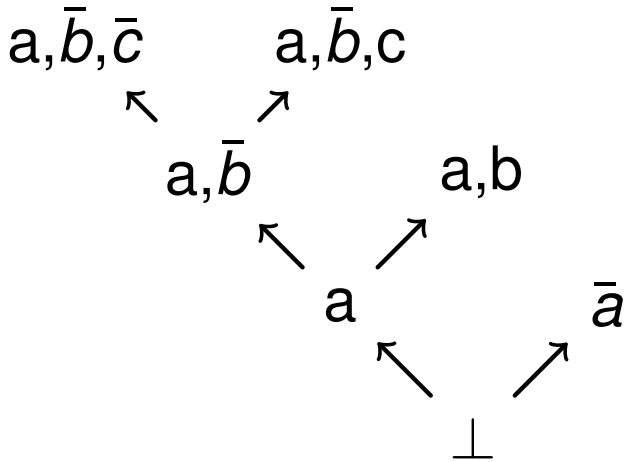
Propositional Example



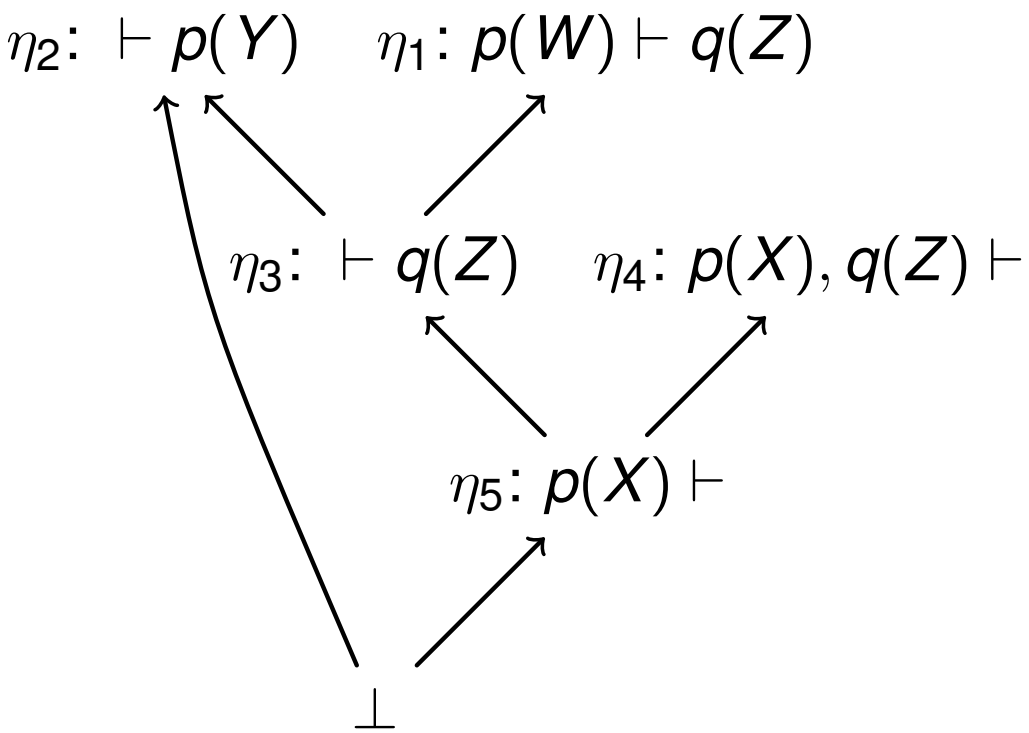


Propositional Example

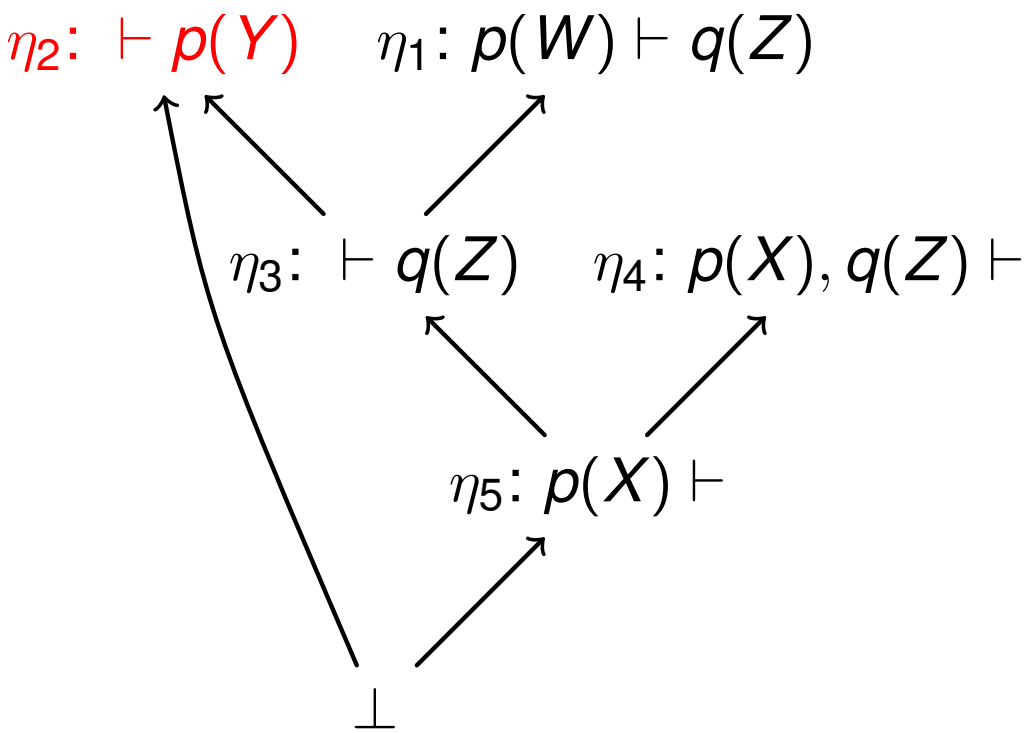




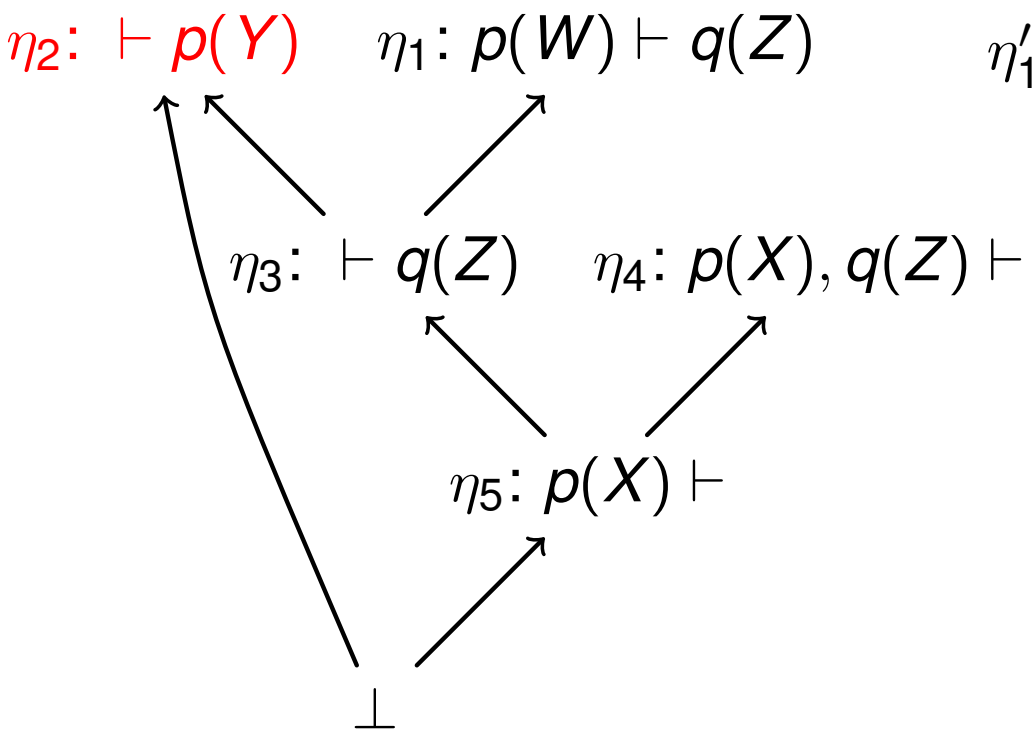
First-Order Change: Helpful Contractions

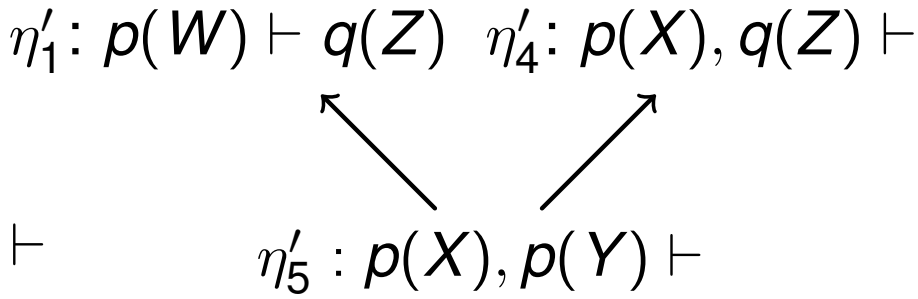


First-Order Change: Helpful Contractions

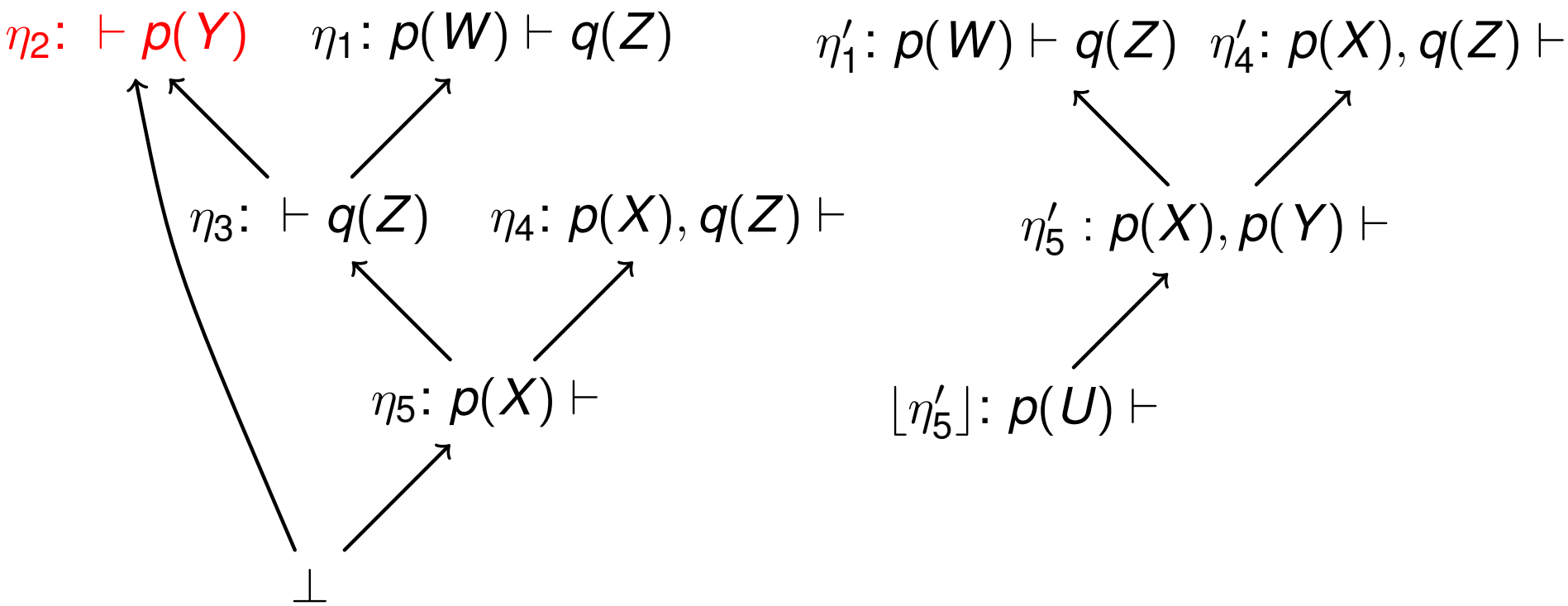


First-Order Change: Helpful Contractions

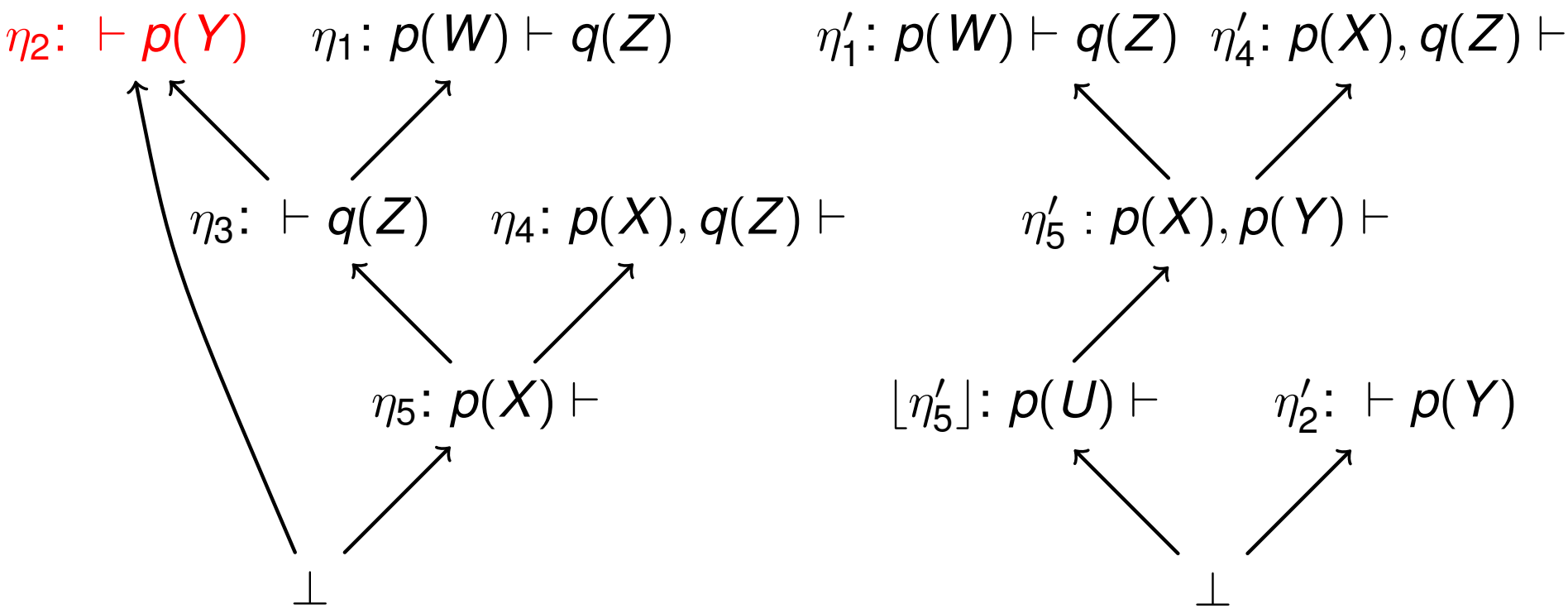




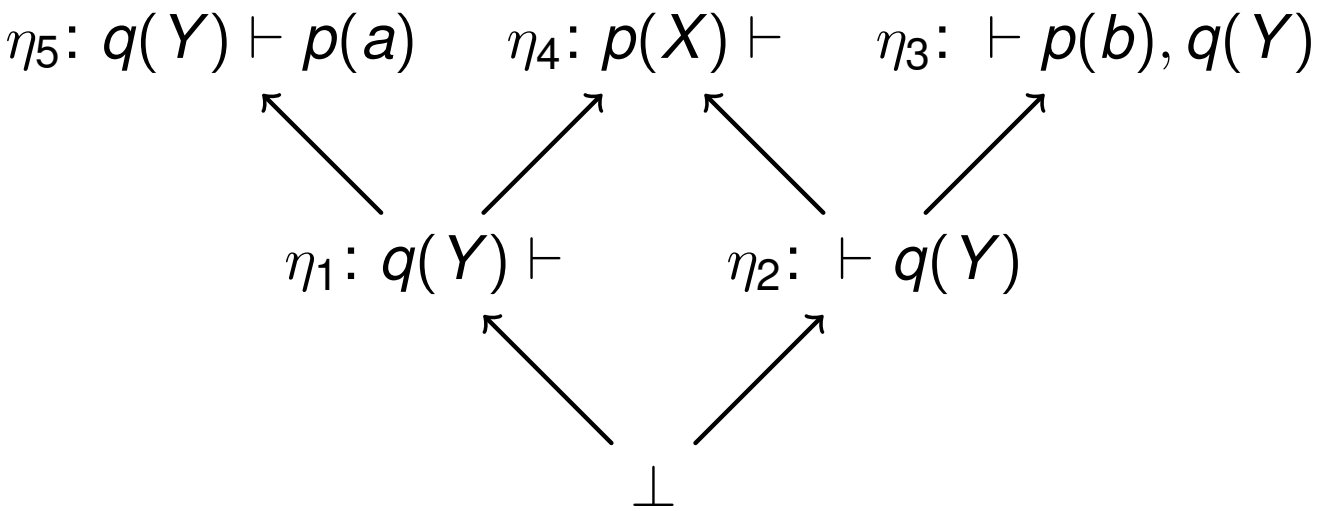
First-Order Change: Helpful Contractions



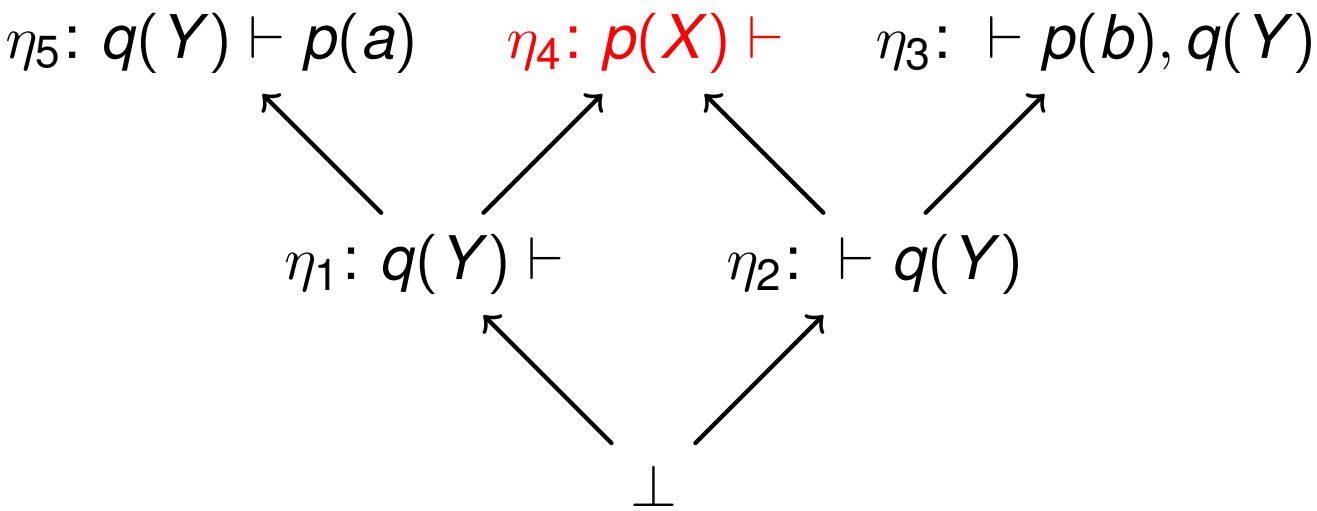
First-Order Change: Helpful Contractions



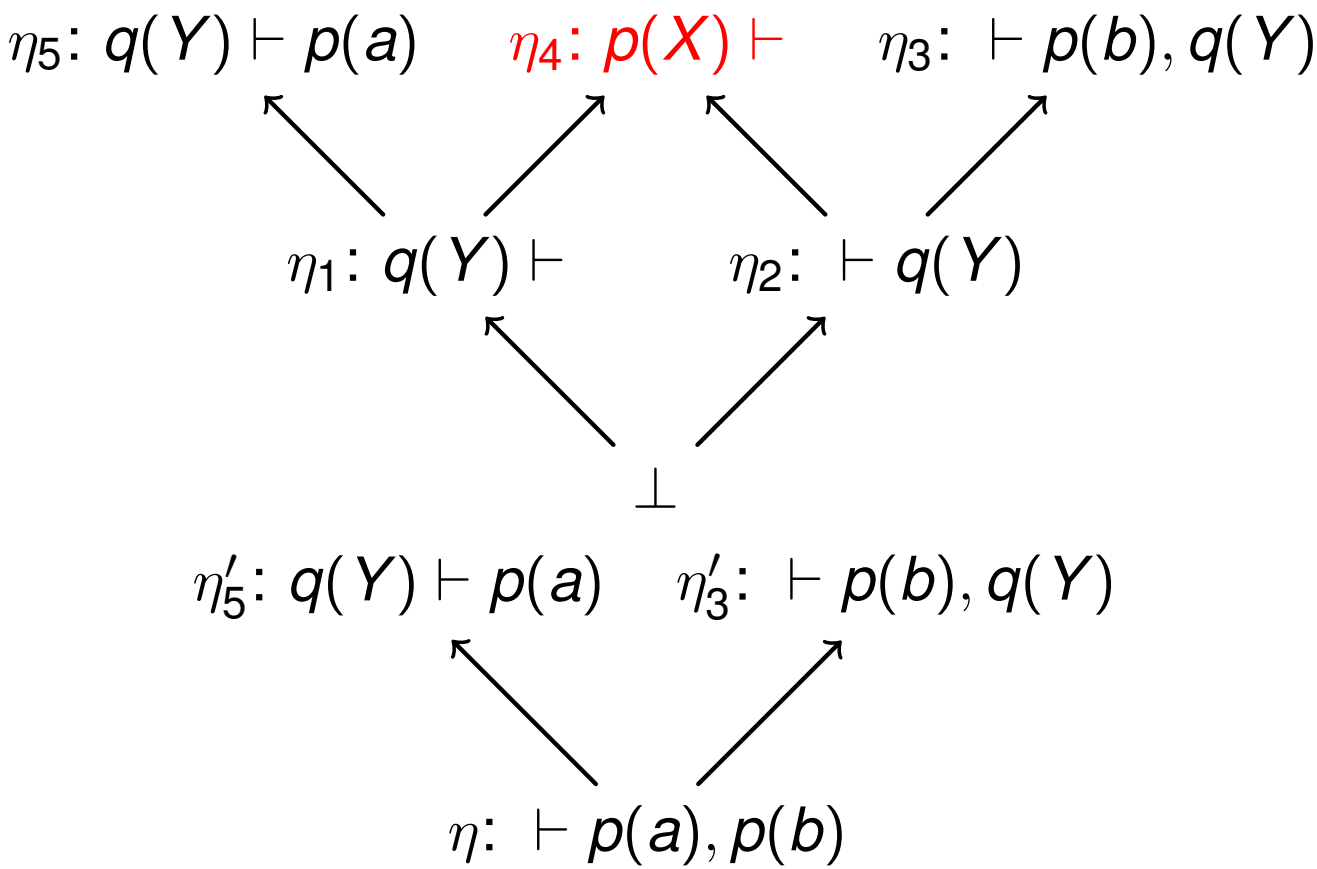
First-Order Challenge: Pre-Deletion Check



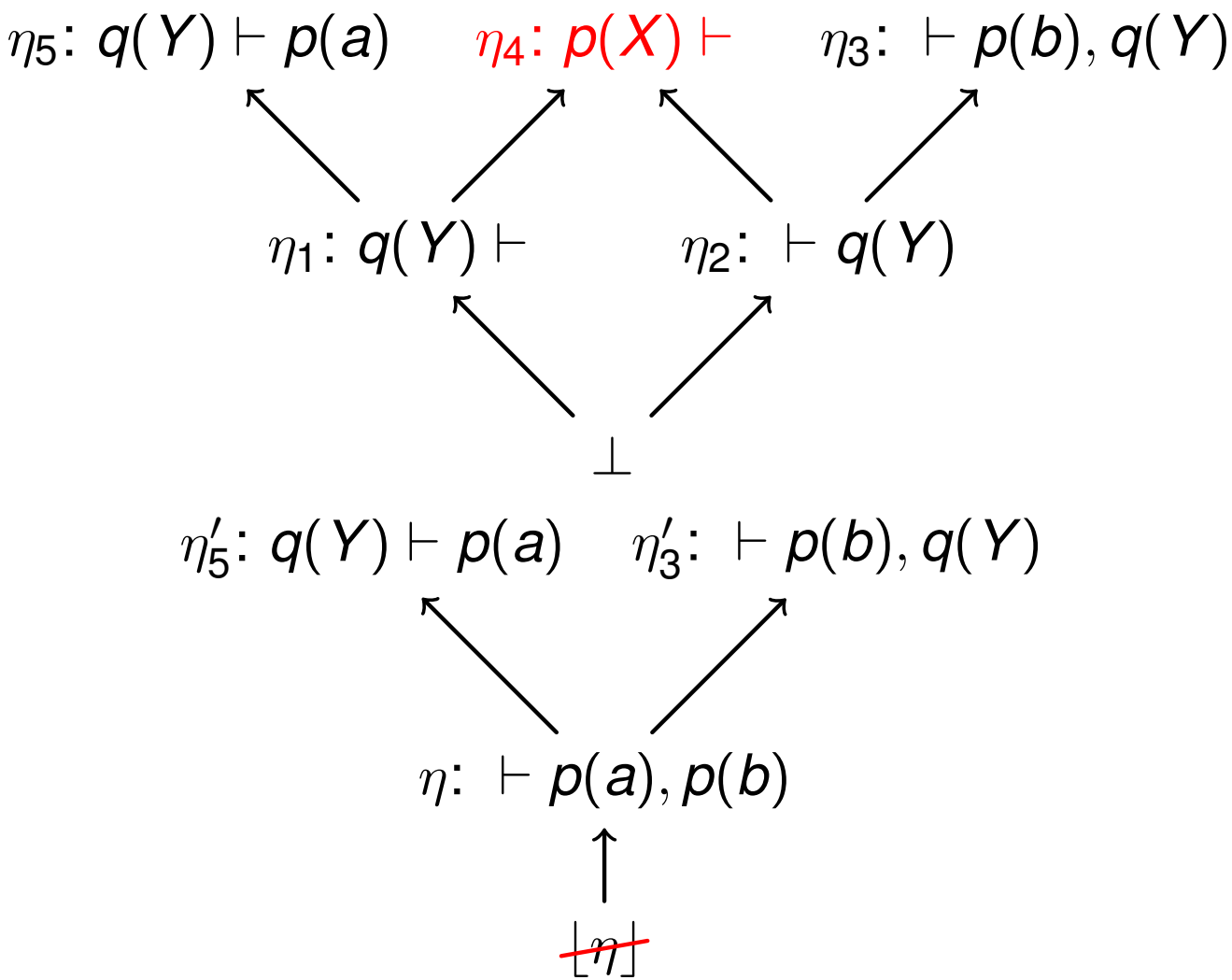
First-Order Challenge: Pre-Deletion Check



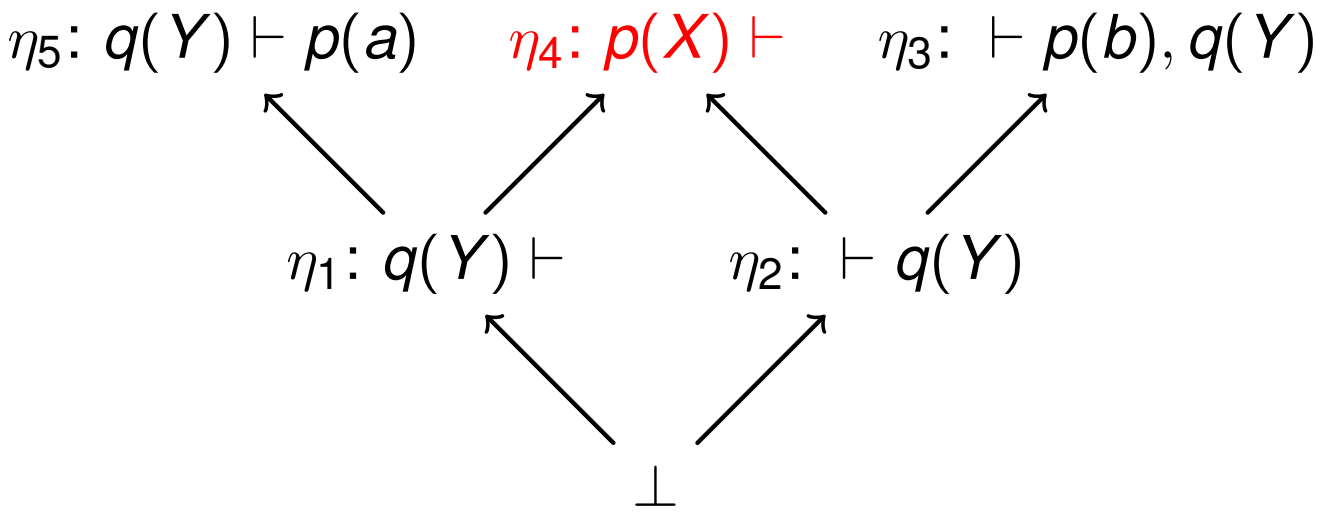
First-Order Challenge: Pre-Deletion Check



First-Order Challenge: Pre-Deletion Check



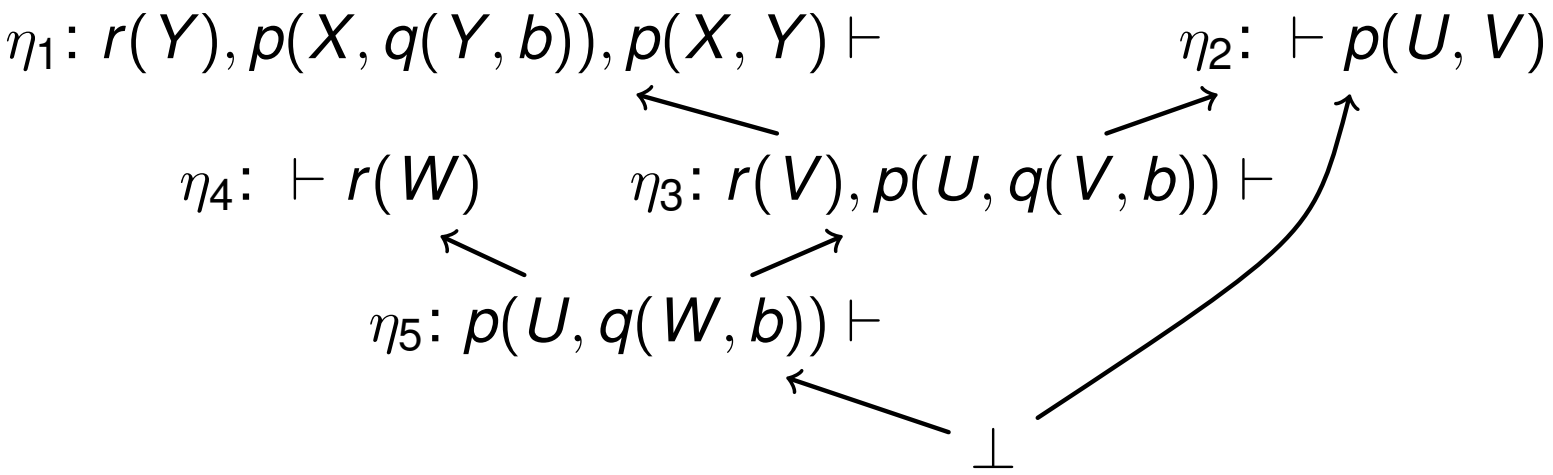
First-Order Challenge: Pre-Deletion Check



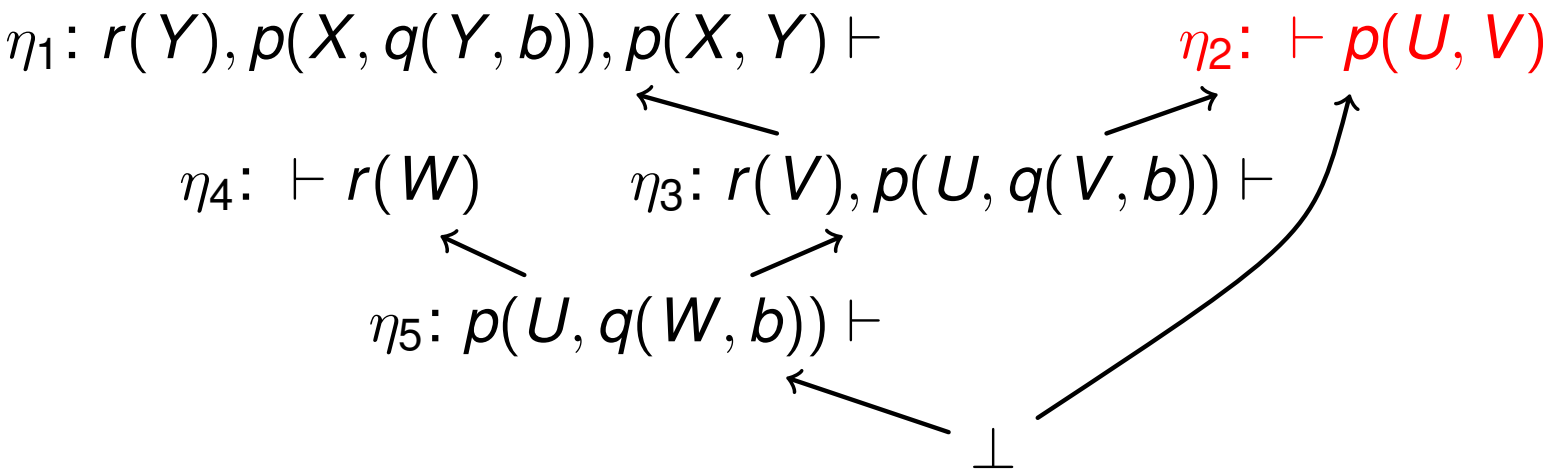
Definition (Pre-Deletion Property)

η unit, $l \in \eta$, such that l is resolved with literals $l_1, \ldots, l_n$ in a proof ψ. η satisfies the *pre-deletion unifiability* property in ψ if $l_1, \ldots, l_n$ and $\bar{l}$ are unifiable.

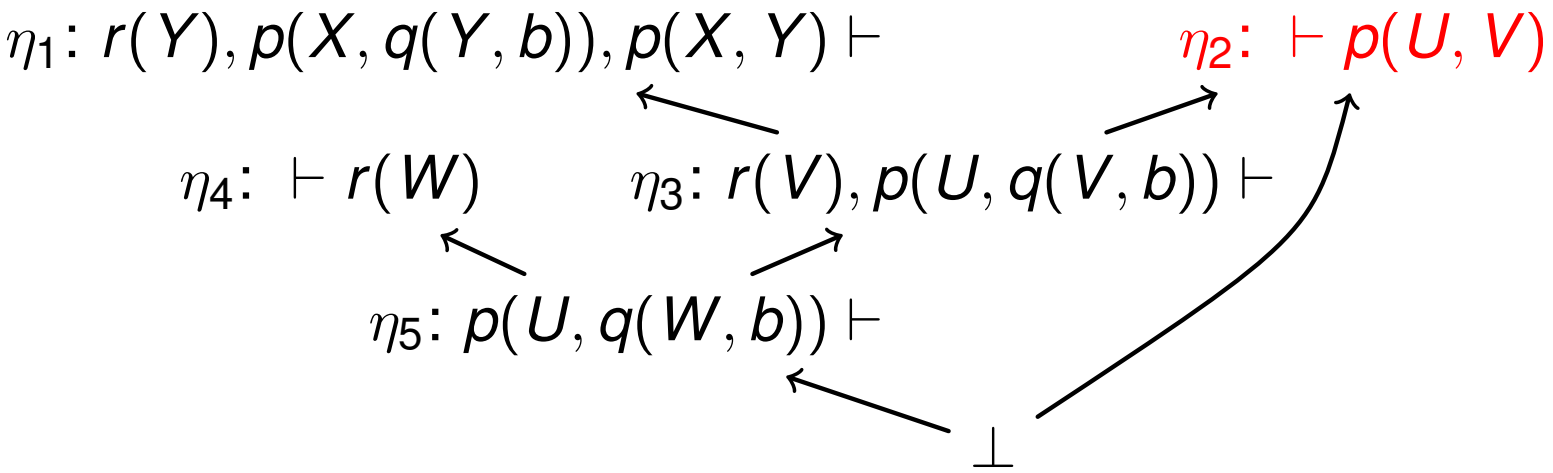
First-Order Challenge: Post-Deletion Check

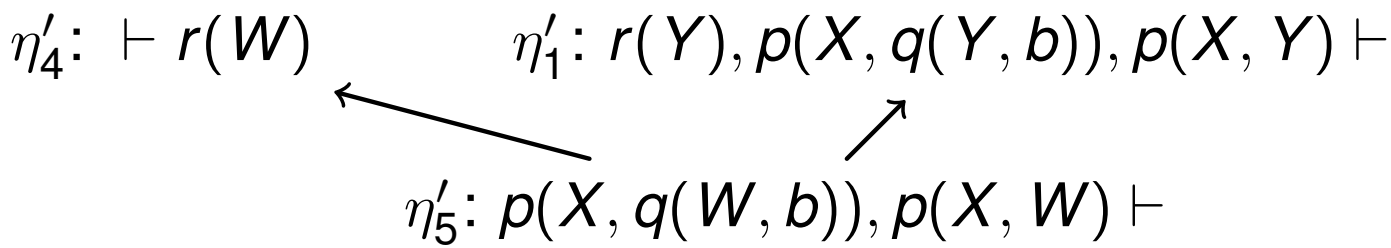


First-Order Challenge: Post-Deletion Check

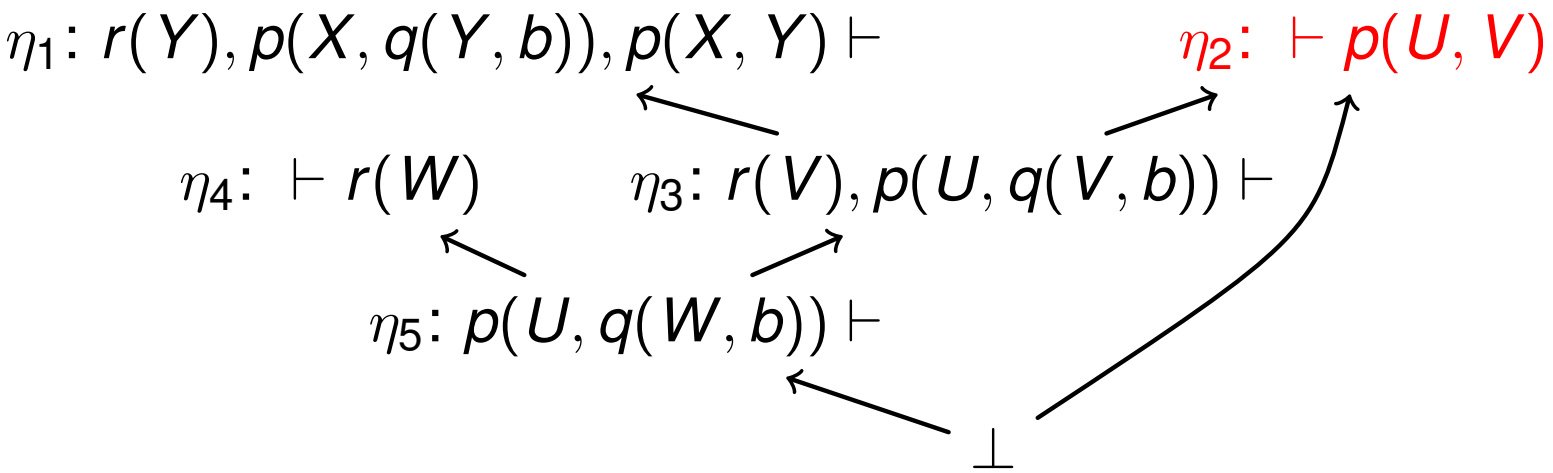


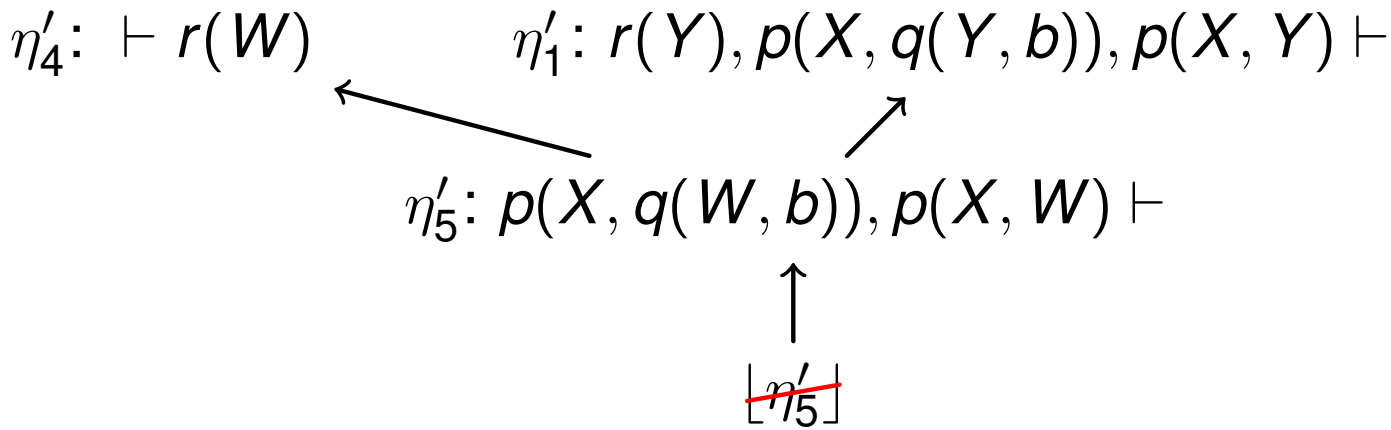
First-Order Challenge: Post-Deletion Check



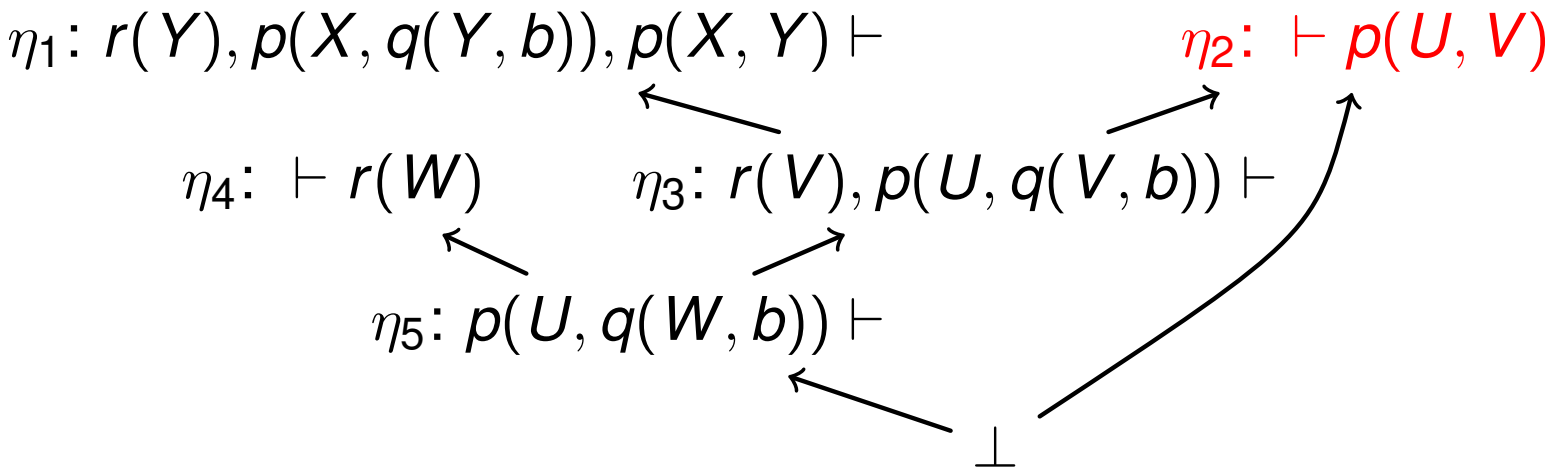


First-Order Challenge: Post-Deletion Check





First-Order Challenge: Post-Deletion Check



Definition (Post-Deletion Property)

η unit, $l \in \eta$, such that l is resolved with literals $l_1, \ldots, l_n$ in a proof ψ. η satisfies the *post-deletion unifiability* property in ψ if $l_1^{\dagger\downarrow}, \ldots, l_n^{\dagger\downarrow}$ and $\overline{l^{\dagger}}$ are unifiable, where $l^{\dagger}$ is the literal in $\psi' = \psi \setminus \{\eta\}$ corresponding to l in ψ, and $l^{\dagger\downarrow}$ is the descendant of $l^{\dagger}$ in the roof of ψ'.

First-Order Lower Units Challenges

- Deletion changes literals
- Unit collection depends on whether contraction is possible after propagation down the proof

Deletion of units require knowledge of proof after deletion, and deletion depends on what will be lowered.

- $O(n^2)$ solution to have full knowledge
- Difficult bookkeeping required for implementation

Greedy First-Order Lower Units - A Quicker Alternative

- Ignore post-deletion satisfaction
- Focus on pre-deletion satisfaction
- Greedy contraction

Faster run-time (linear; one traversal)
Easier to implement

- Doesn't always compress (returns original proof sometimes)

Greedy First-Order Lower Units - A Quicker Alternative

- Ignore post-deletion satisfaction
- Focus on pre-deletion satisfaction
- Greedy contraction

Faster run-time (linear; one traversal)
Easier to implement

- Doesn't always compress (returns original proof sometimes)

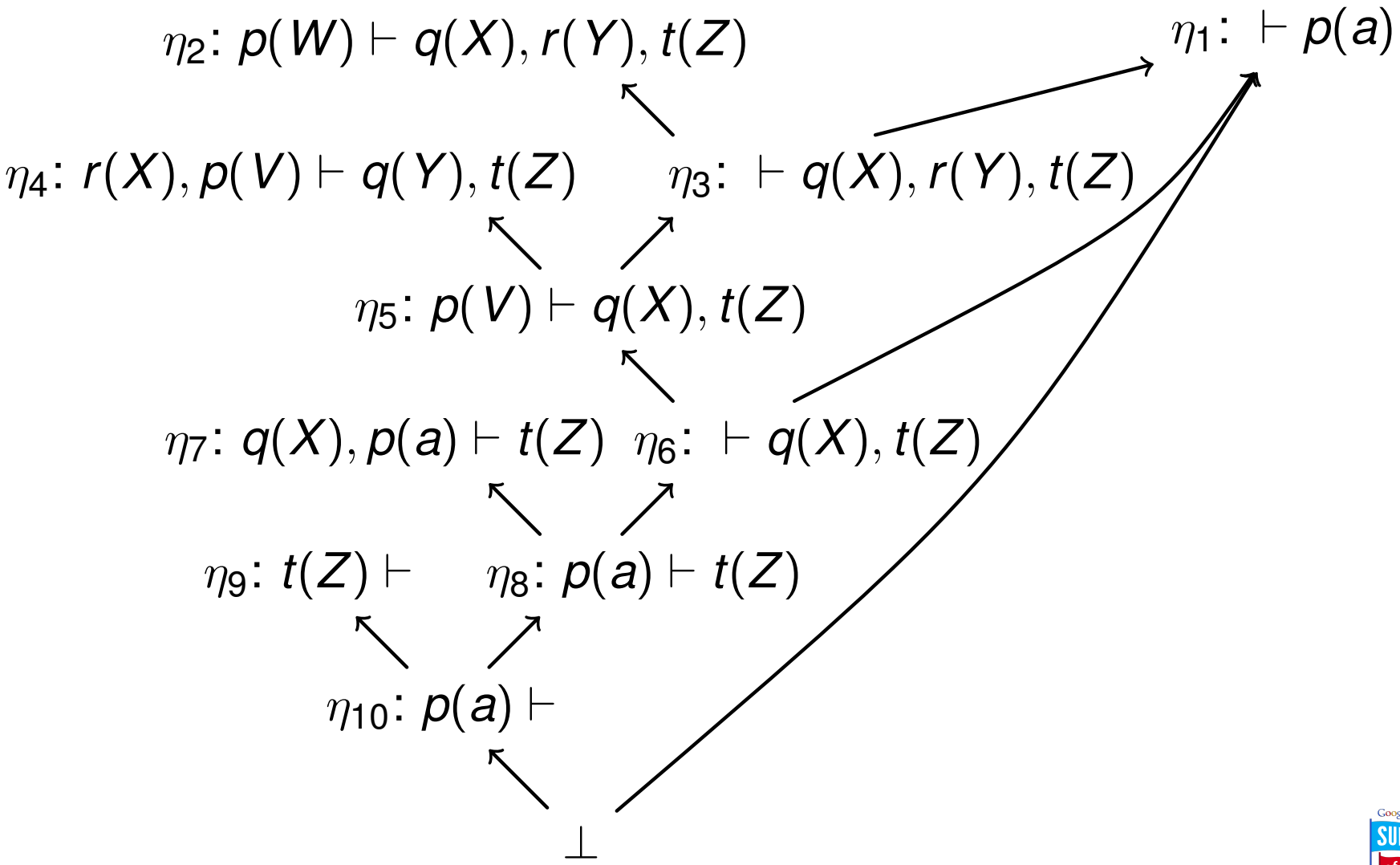
Greedy First-Order Lower Units - A Quicker Alternative

- Ignore post-deletion satisfaction
- Focus on pre-deletion satisfaction
- Greedy contraction

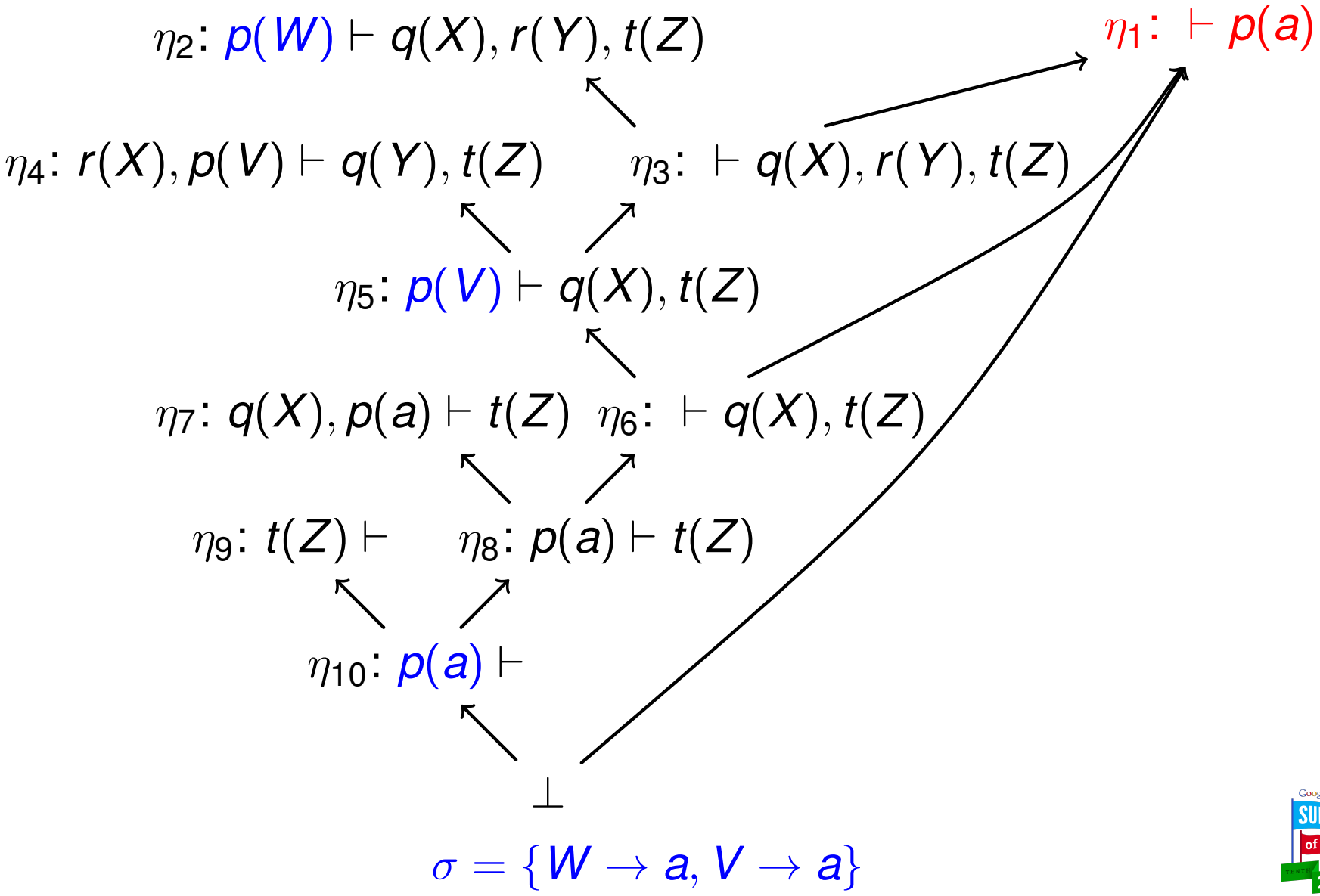
Faster run-time (linear; one traversal)
Easier to implement

- Doesn't always compress (returns original proof sometimes)

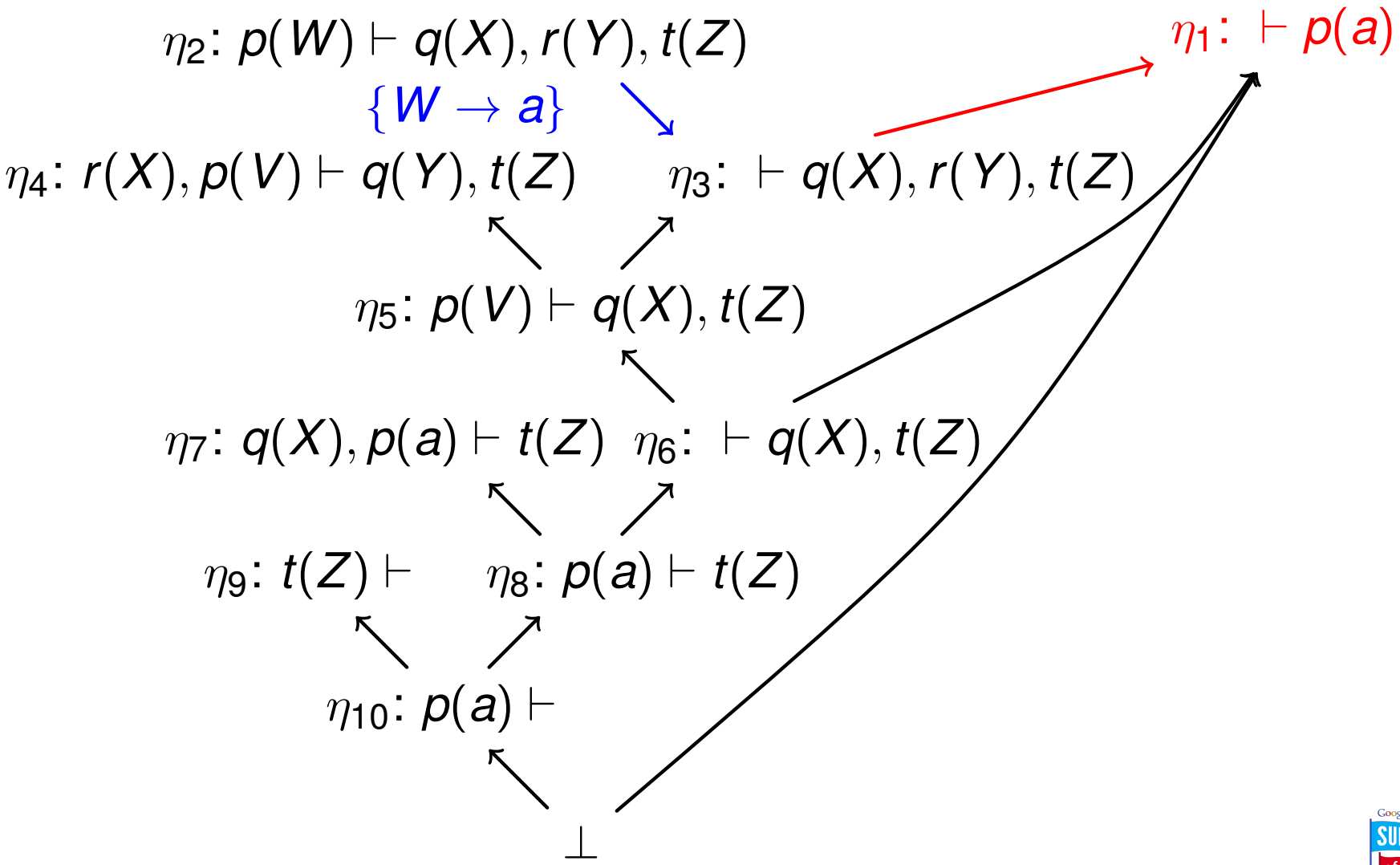
First-Order Example



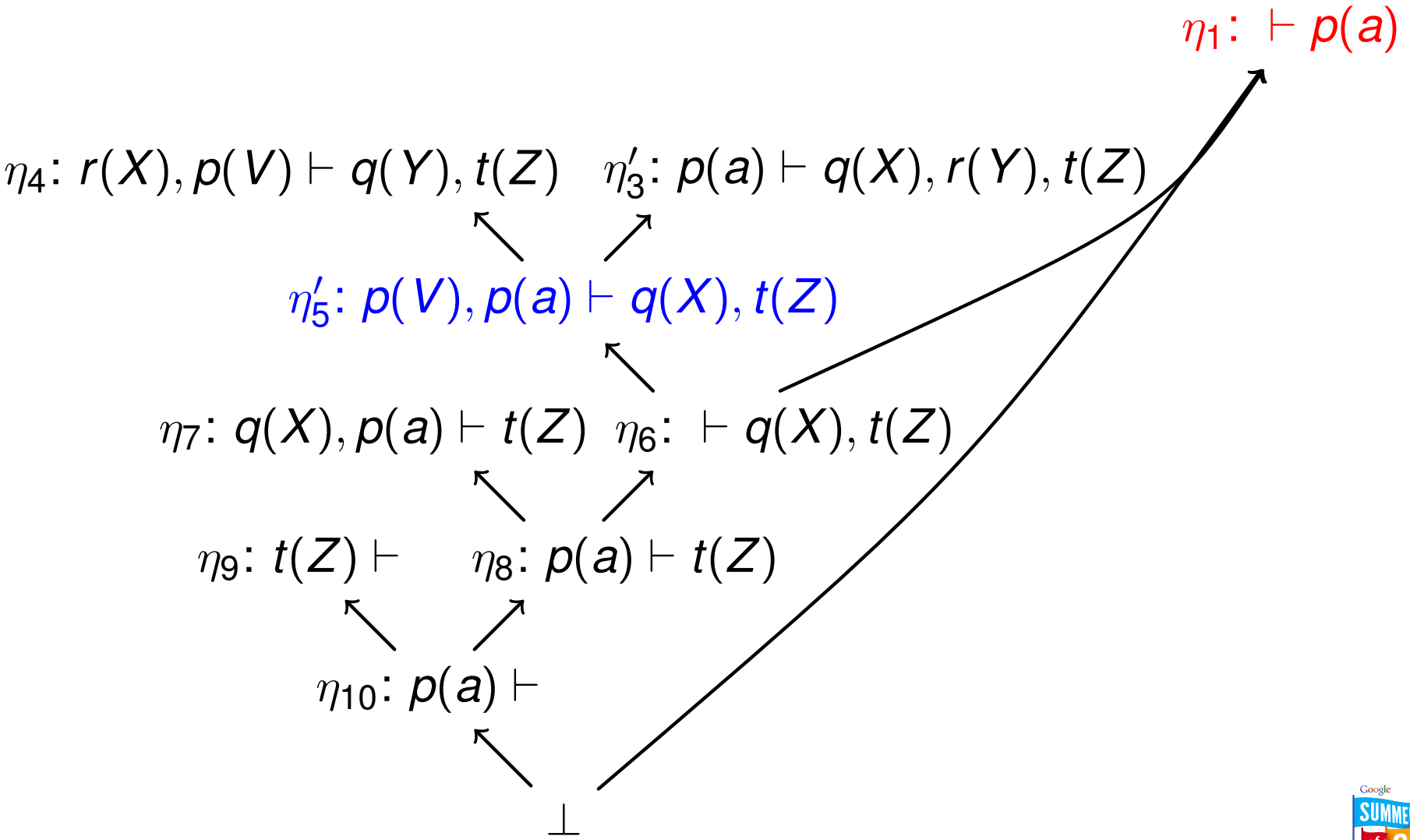
First-Order Example



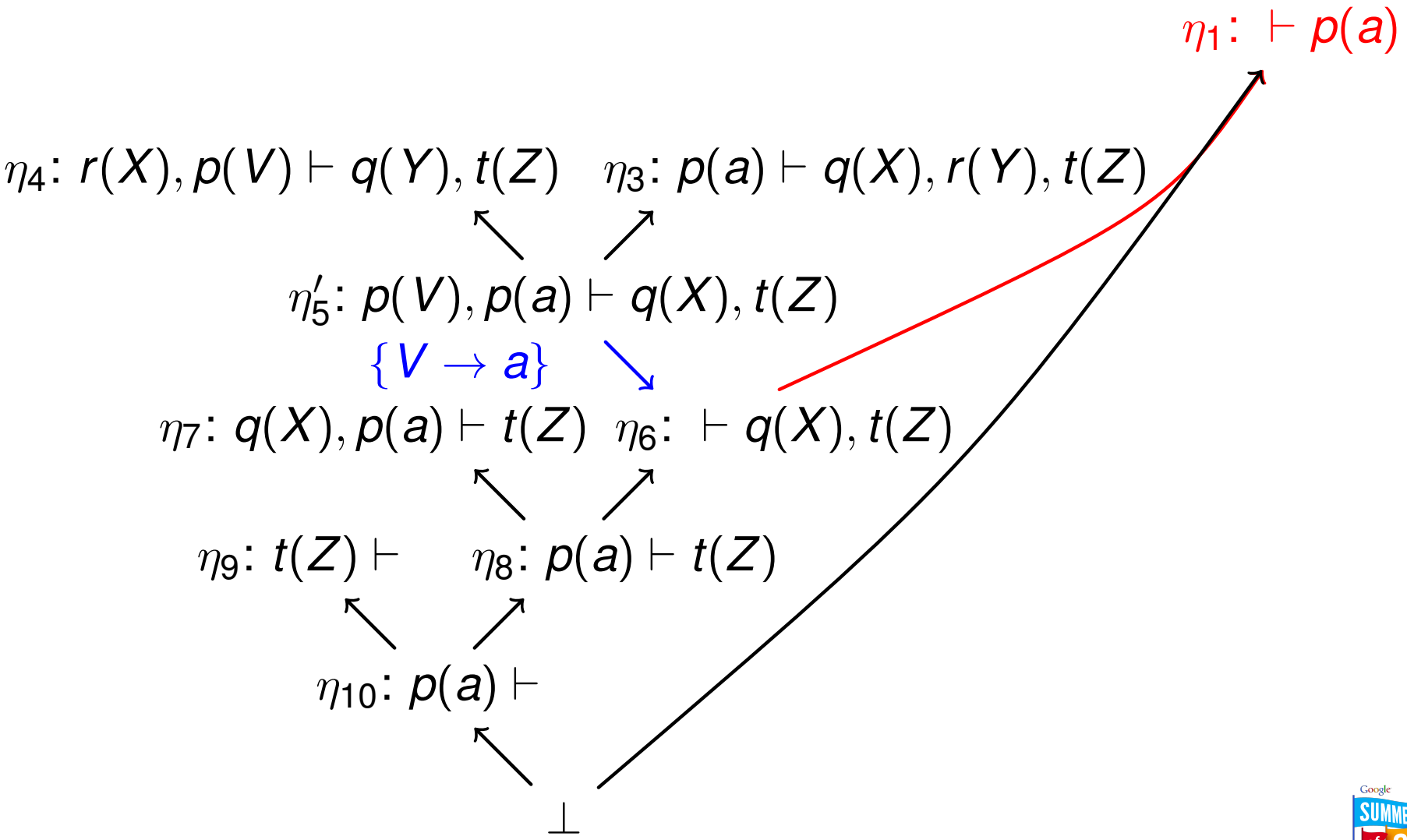
First-Order Example



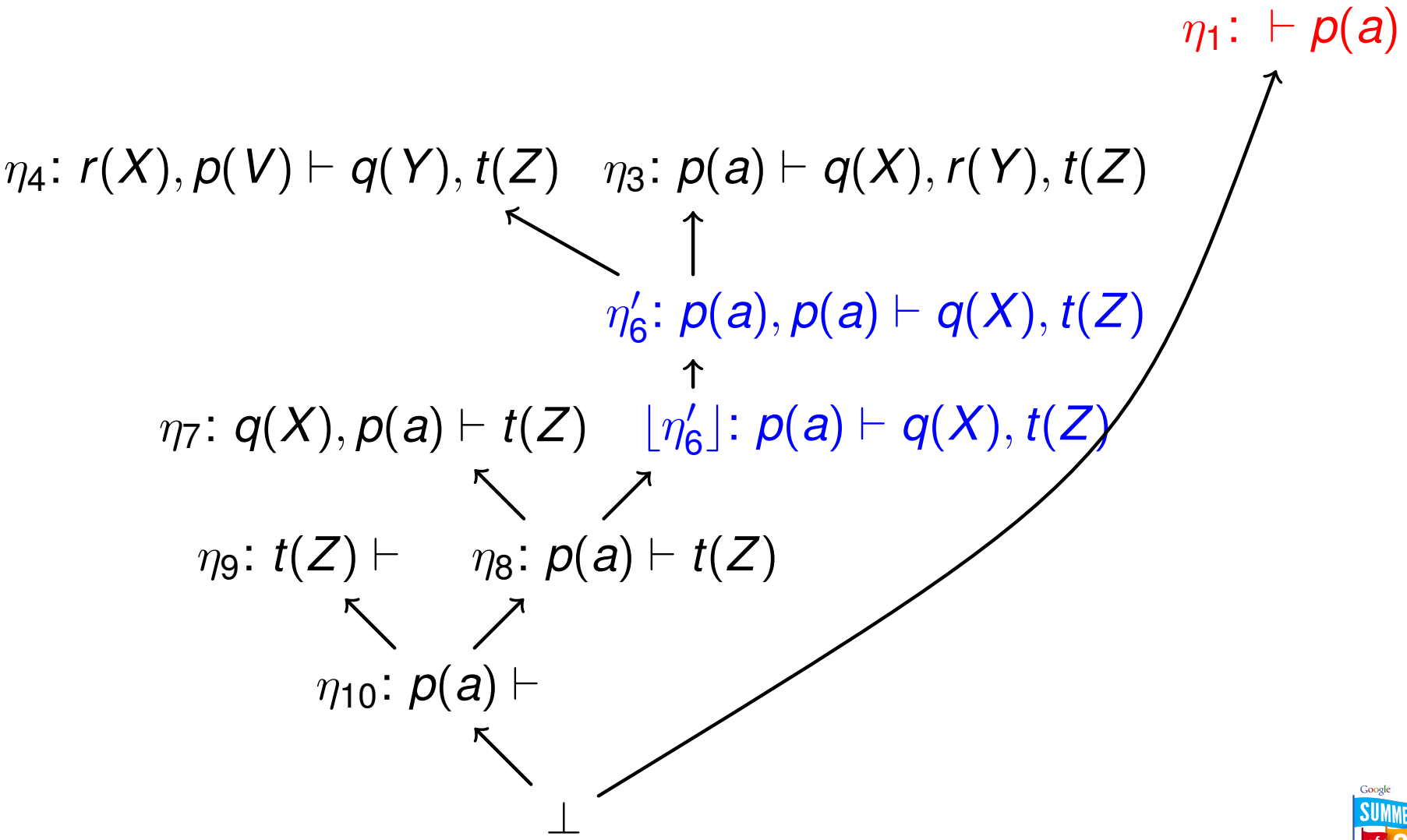
First-Order Example



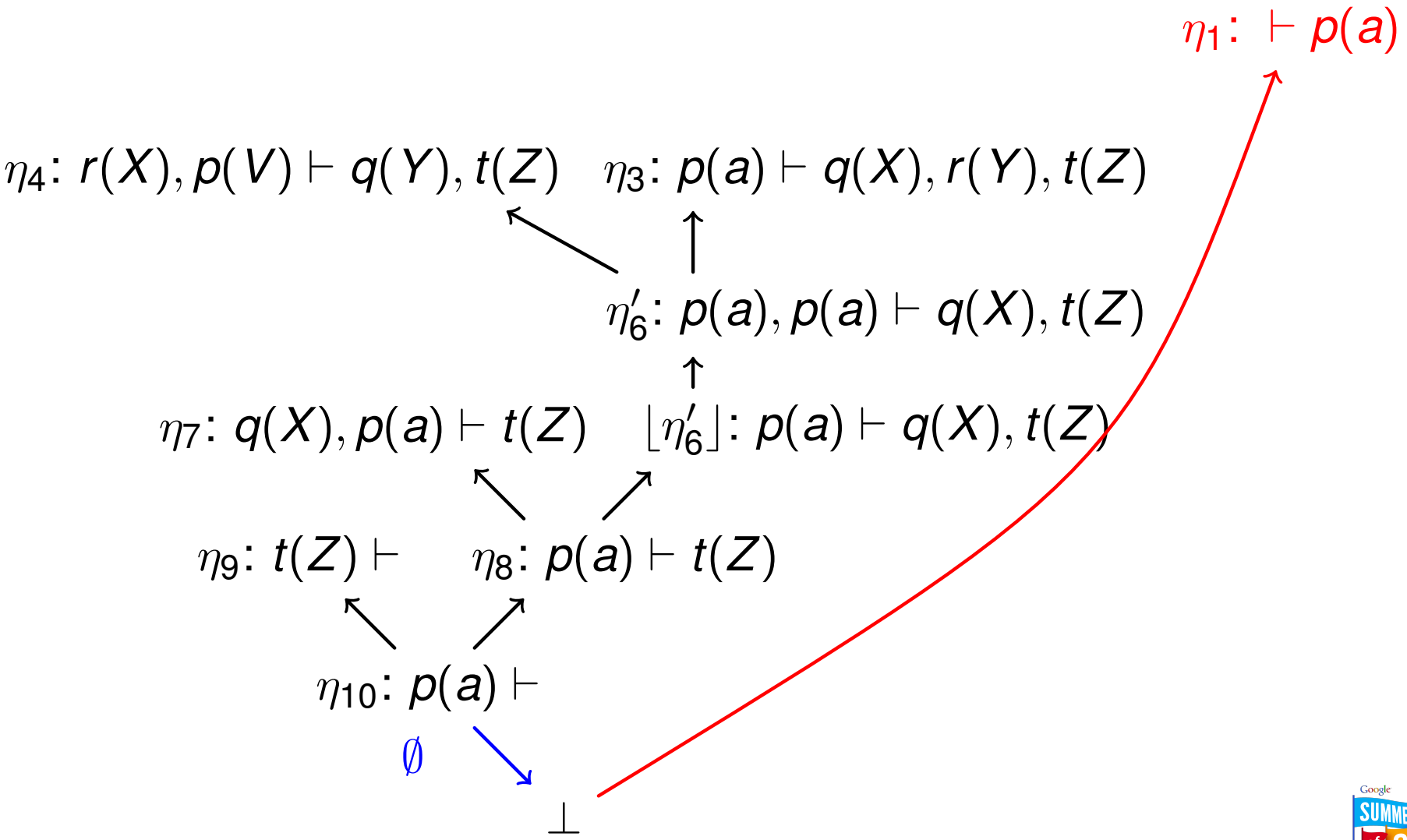
First-Order Example



First-Order Example

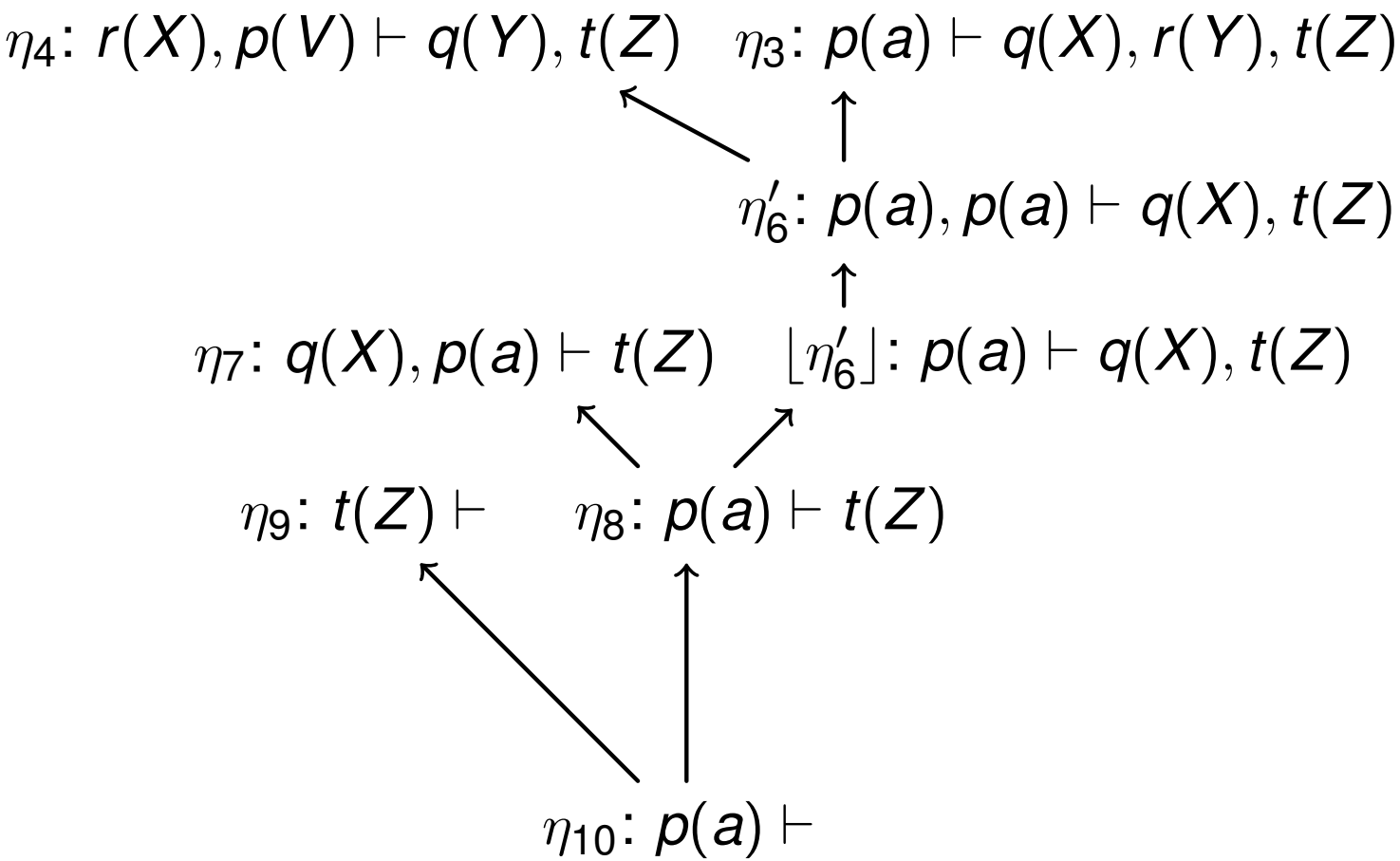


First-Order Example

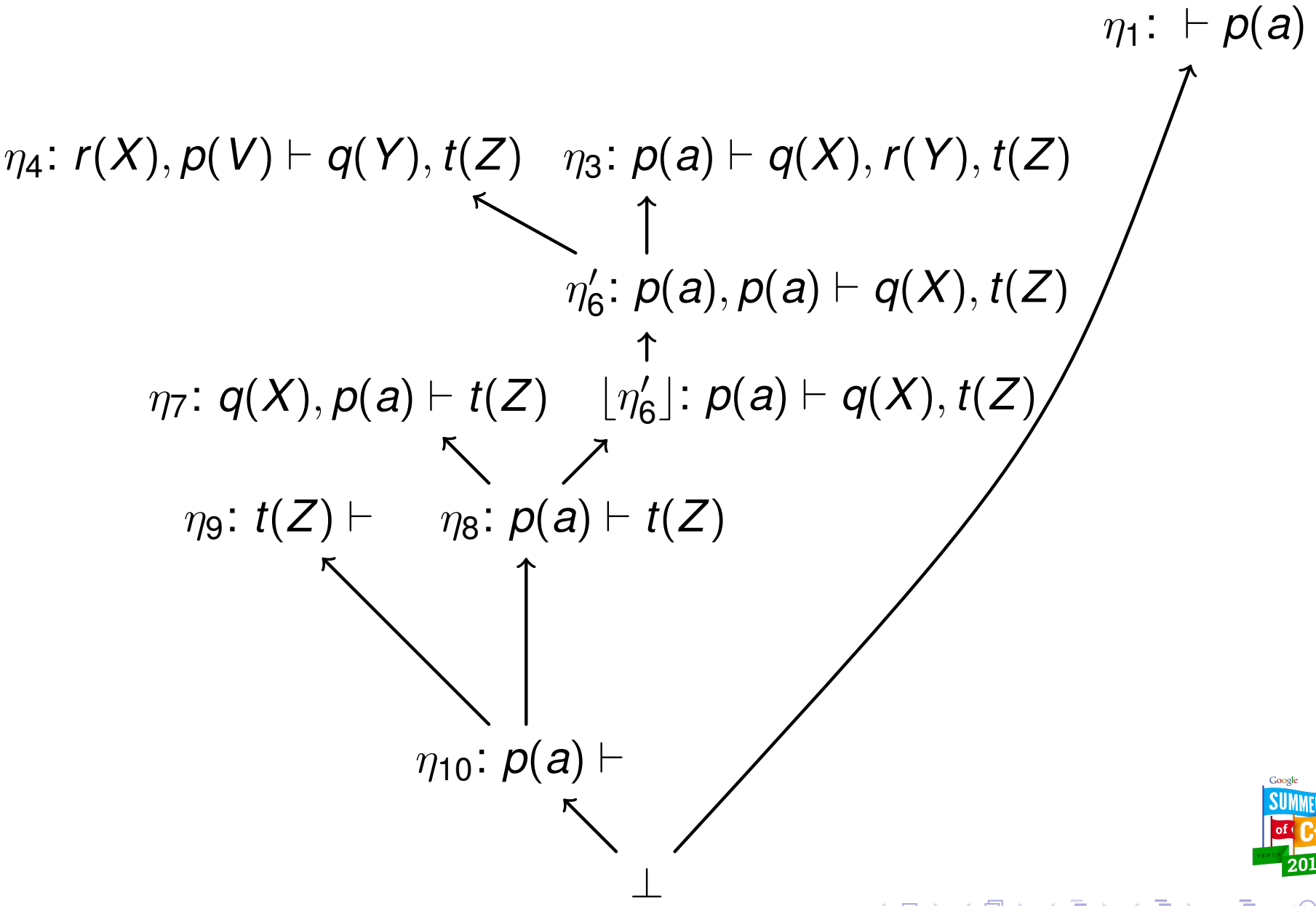


First-Order Example

$\eta_1 \colon \vdash p(a)$



First-Order Example



Experiment Setup

- Simple First-Order Lower units implemented as part of Skeptik (in Scala)
- 308 real first-order proofs generated by SPASS from problems from TPTP Problem Library
  - 2280 initial problems (1032 known unsatisfiable)
  - SPASS asked to use only resolution and contraction rules
  - 300s timeout
- proofs *generated* on cluster at the University of Victoria
- proofs *compressed* on *this laptop*

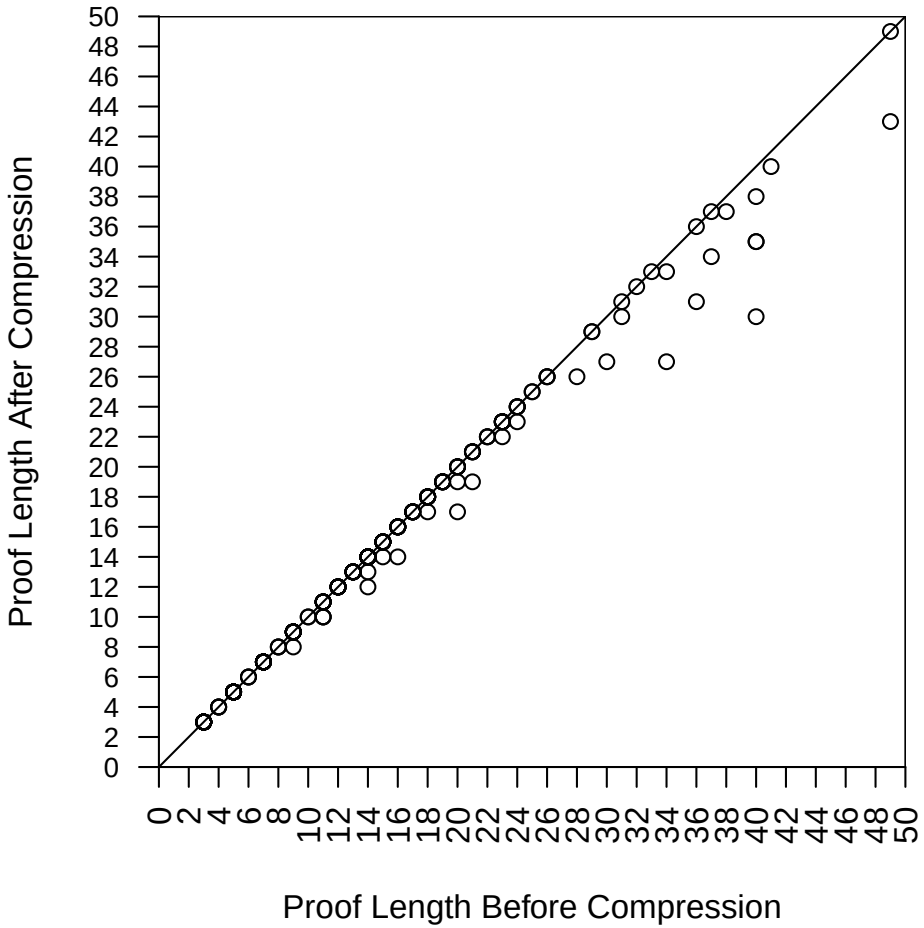
Time to generate proofs: $\approx$ 40 minutes
Time to compress proofs: $\approx$ 5 seconds

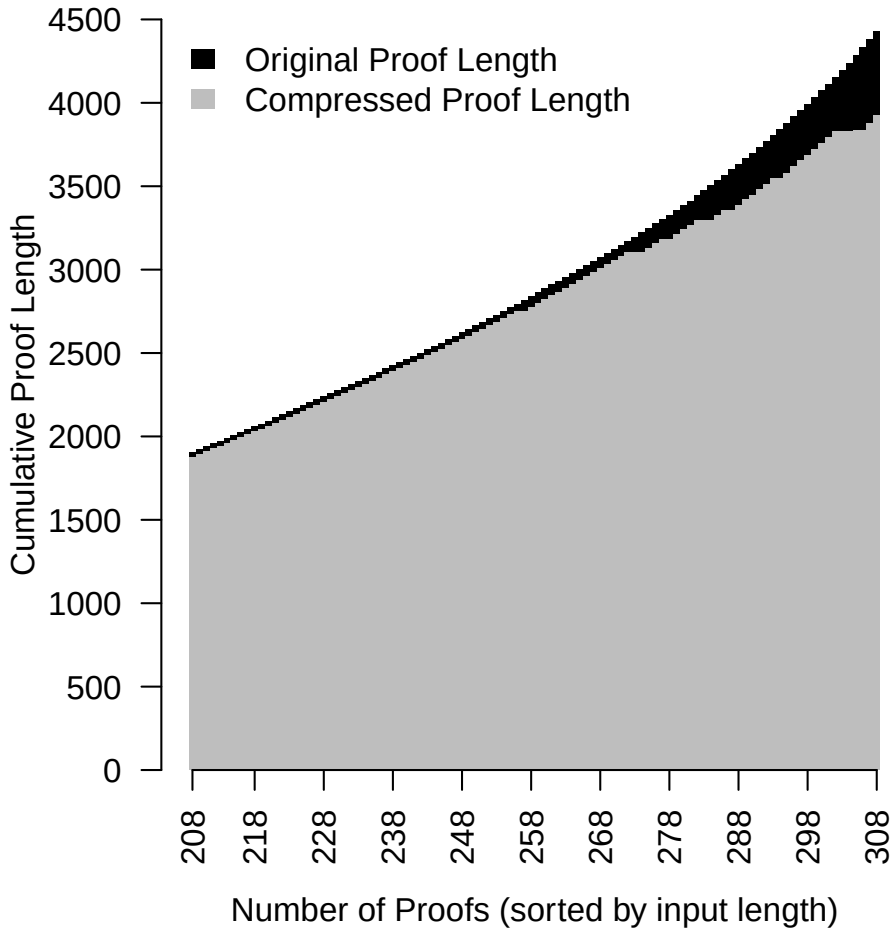
Experiment Setup

- Simple First-Order Lower units implemented as part of Skeptik (in Scala)
- 308 real first-order proofs generated by SPASS from problems from TPTP Problem Library
  - 2280 initial problems (1032 known unsatisfiable)
  - SPASS asked to use only resolution and contraction rules
  - 300s timeout
- proofs *generated* on cluster at the University of Victoria
- proofs *compressed* on *this laptop*

Time to generate proofs: $\approx$ 40 minutes
Time to compress proofs: $\approx$ 5 seconds

Results


Proof Length After Compression
Proof Length Before Compression
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
40
42
44
46
48
50


Original Proof Length
Compressed Proof Length
Cumulative Proof Length
Number of Proofs (sorted by input length)
0
500
1000
1500
2000
2500
3000
3500
4000
4500
208
218
228
238
248
258
268
278
288
298
308

Results

Higher compression in longer proofs:
13/18 proofs with length ≥ 30 nodes successfully compressed.

Total compression ratio 11.3%: 4429 vs. 3929 nodes.
18.4% for 100 longest proofs.

Only 14/308 proofs failed to satisfy the post-deletion unifiability property

Conclusion

- Simple First-Order Lower Units is a quick algorithm for first-order proof compression
- Future work:
  - Explore other proof compression algorithms, e.g. Recycle Pivots with Intersection
  - Explore ways of dealing with the post-deletion property quickly

Thank you for your attention.
Any questions?

- Source code: `https://github.com/jgorzny/Skeptik`
- Data: `http://www.math.uvic.ca/~jgorzny/data/`