Scalable, Interpretable Protocol Verification by Inductive Proof Slicing

William Schultz*, Eddy Ashton†, Heidi Howard†, Stavros Tripakis*
Northeastern University*
Microsoft Research Cambridge†

New England Systems Verification Day 2024

Plethora of new distributed protocol verification techniques in past several years.

IC3PO [NFM21,FMCAD21]

FOL-IC3 [PLDI20]

SWISS [NSDI21]

endive [FMCAD22]

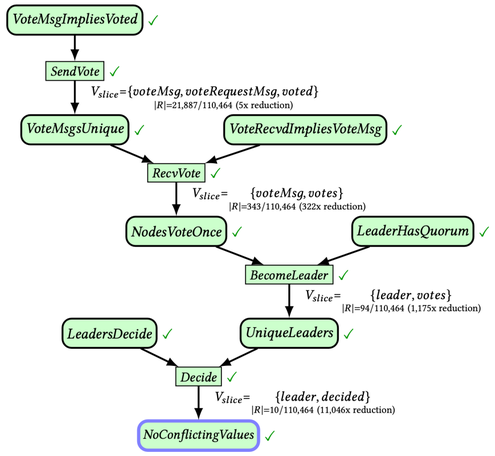
DistAI [OSDI21, OSDI22]

Core task: **inductive invariant synthesis**.

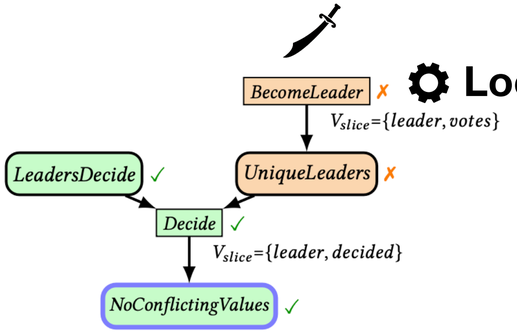
For tackling larger protocol verification tasks, we want both better ***scalability*** and ***interpretability****.*

Our Work

1. Inductive Proof Graph



2. Inductive Proof Slicing



Localized Slicing + Lemma Synthesis

Inductive Proof Graph

Running example: Simple consensus protocol.

6 state variables

5 concurrent actions.

CONSTANTS $Node, Value, Quorum$

VARIABLES
$voteRequestMsg,$
$voted,$
$voteMsg,$
$votes,$
$leader,$
$decided$

$Init \triangleq$ Initial states.
$\land\ voteRequestMsg = \{\}$
$\land\ voted = [i \in Node \mapsto \text{False}]$
$\land\ voteMsg = \{\}$
$\land\ votes = [i \in Node \mapsto \{\}]$
$\land\ leader = [i \in Node \mapsto \text{False}]$
$\land\ decided = [i \in Node \mapsto \{\}]$

$Next \triangleq$ Transition relation.
$\exists\, i, j \in Node :$
$\exists\, v \in Value :$
$\exists\, Q \in Quorum :$
$\lor\ SendRequestVote(i, j)$
$\lor\ SendVote(i, j)$
$\lor\ RecvVote(i, j)$
$\lor\ BecomeLeader(i, Q)$
$\lor\ Decide(i, v)$

Protocol actions.

$SendRequestVote(src, dst) \triangleq$
$\land\ voteRequestMsg' = voteRequestMsg \cup \{\langle src, dst\rangle\}$

$SendVote(src, dst) \triangleq$
$\land\ \neg voted[src]$
$\land\ \langle dst, src\rangle \in voteRequestMsg$
$\land\ voteMsg' = voteMsg \cup \{\langle src, dst\rangle\}$
$\land\ voted'[src] := \text{True}$
$\land\ voteRequestMsg' = voteRequestMsg \setminus \{\langle src, dst\rangle\}$

$RecvVote(n, sender) \triangleq$
$\land\ \langle sender, n\rangle \in voteMsg$
$\land\ votes'[n] := votes[n] \cup \{sender\}$

$BecomeLeader(n, Q) \triangleq$
$\land\ Q \subseteq votes[n]$
$\land\ leader'[n] := \text{True}$

$Decide(n, v) \triangleq$
$\land\ leader[n]$
$\land\ decided[n] = \{\}$
$\land\ decided'[n] := \{v\}$

Safety property.

$NoConflictingValues \triangleq$
$\forall n_1, n_2 \in Node, v_1, v_2 \in Value :$
$(v_1 \in decided[n_1] \land v_2 \in decided[n_2]) \Rightarrow (v_1 = v_2)$

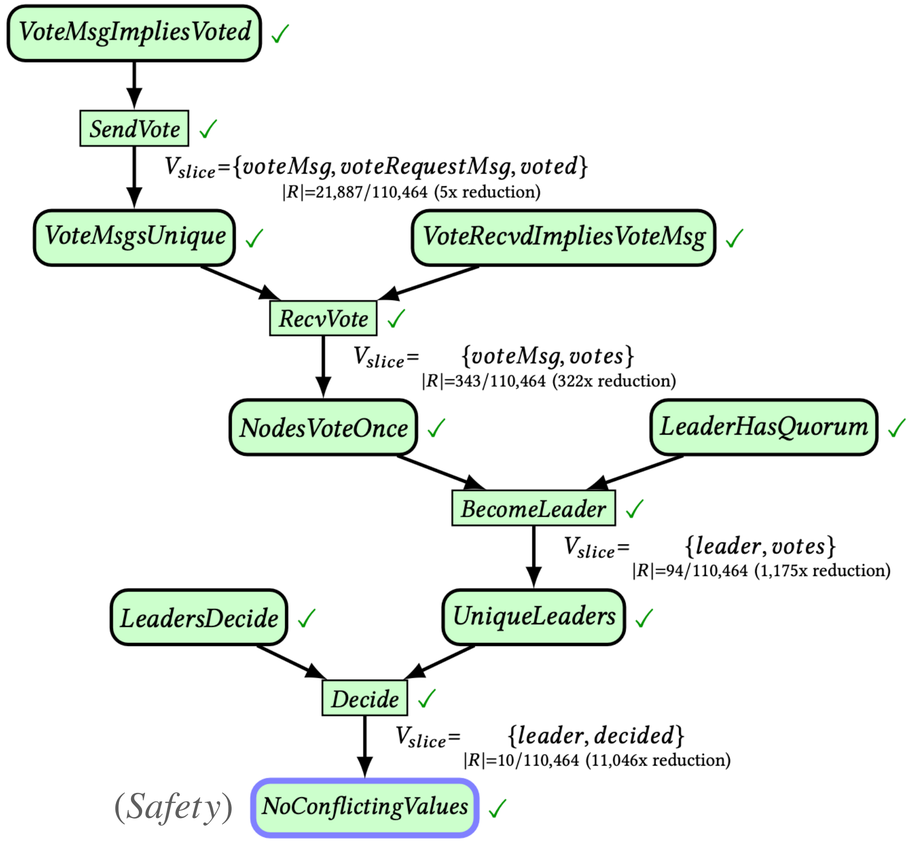
Inductive Proof Graph

$Ind \triangleq$ Inductive invariant.

$\wedge$ *NoConflictingValues* (Safety)
$\wedge$ *UniqueLeaders*
$\wedge$ *LeaderHasQuorum*
$\wedge$ *LeadersDecide*
$\wedge$ *NodesVoteOnce*
$\wedge$ *VoteRecordedImpliesVoteMsg*
$\wedge$ *VoteMsgsUnique*
$\wedge$ *VoteMsgImpliesNodeVoted*

Monolithic inductive invariant for a simple consensus protocol. (conjunction of lemma invariants)

Inductive Decomposition



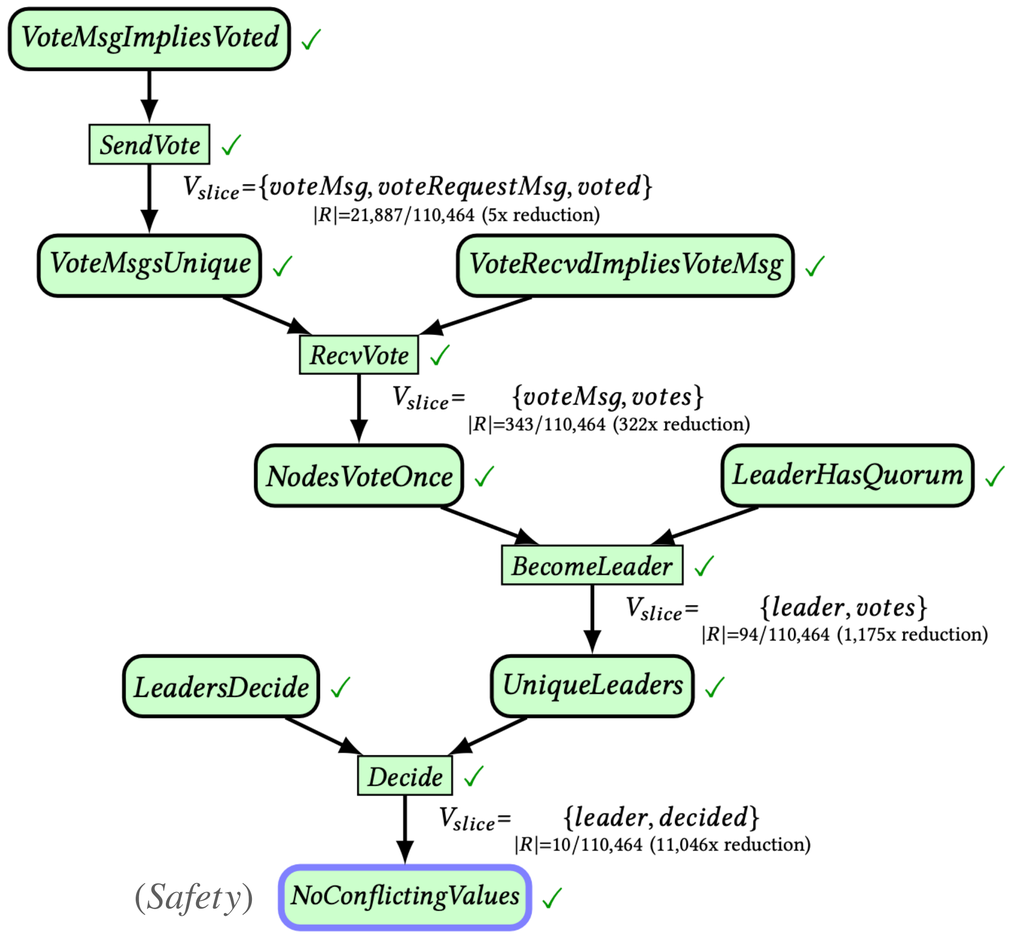
Corresponding inductive proof graph for *Ind*.

Inductive Invariant Synthesis by Inductive Proof Slicing

- Syntax-guided inductive invariant synthesis algorithm built on the inductive proof graph.
- Compute ***variable slices*** at local graph nodes, enabling synthesis acceleration via:
    - **Grammar slicing**
    - **State slicing**

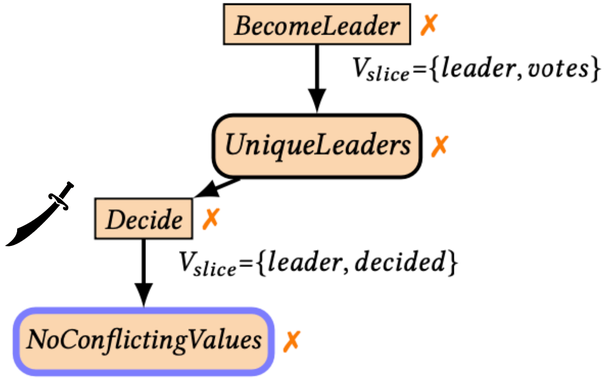
6 protocol state variables

$\{voted, voteMsg, votes, voteRequestMsg, leader, decided\}$



Inductive Invariant Synthesis by Inductive Proof Slicing

- Construct the inductive proof graph incrementally, backwards from safety property.



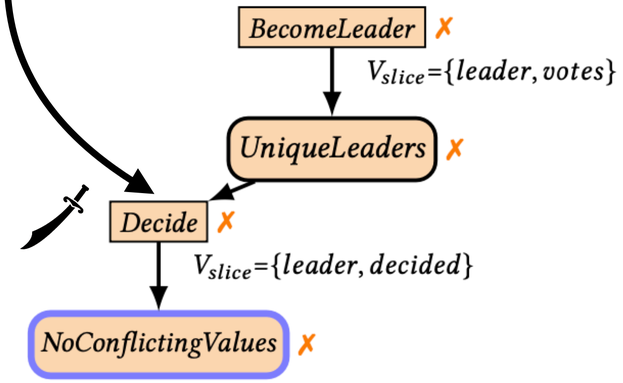
In-progress inductive proof graph.

Inductive Invariant Synthesis by Inductive Proof Slicing

- Construct the inductive proof graph incrementally, backwards from safety property.

Local synthesis of
inductive support lemmas with slicing.

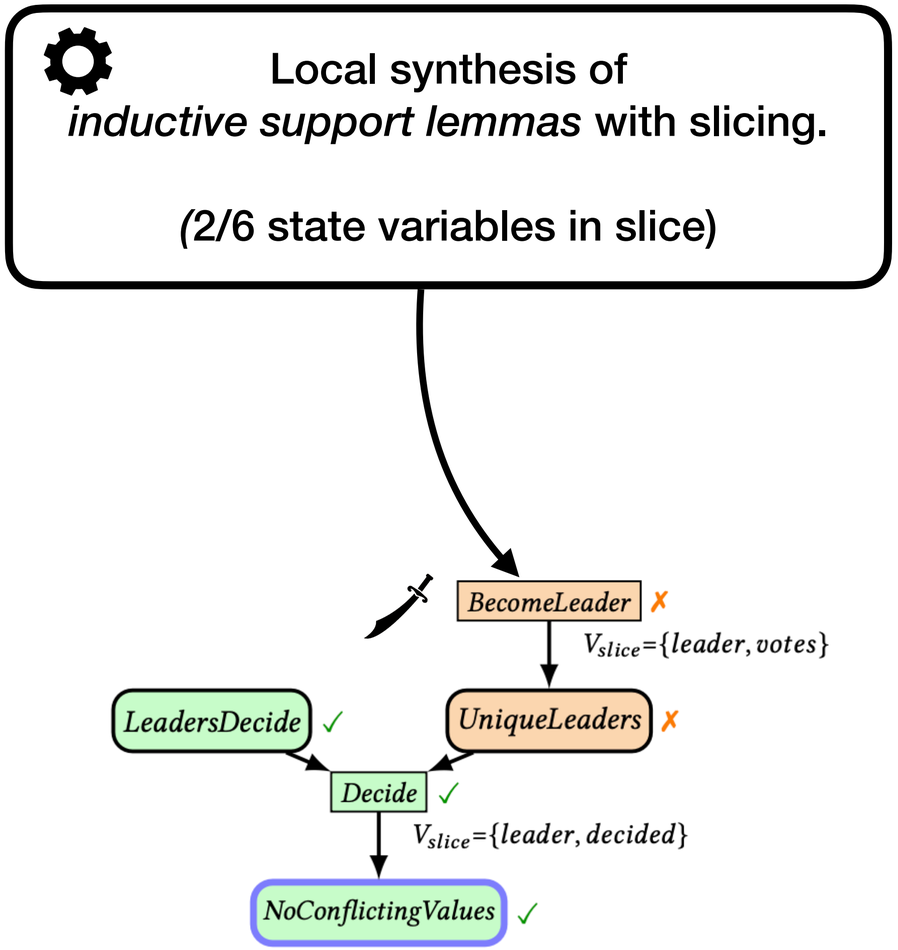
(2/6 state variables in slice)



In-progress inductive proof graph.

Inductive Invariant Synthesis by Inductive Proof Slicing

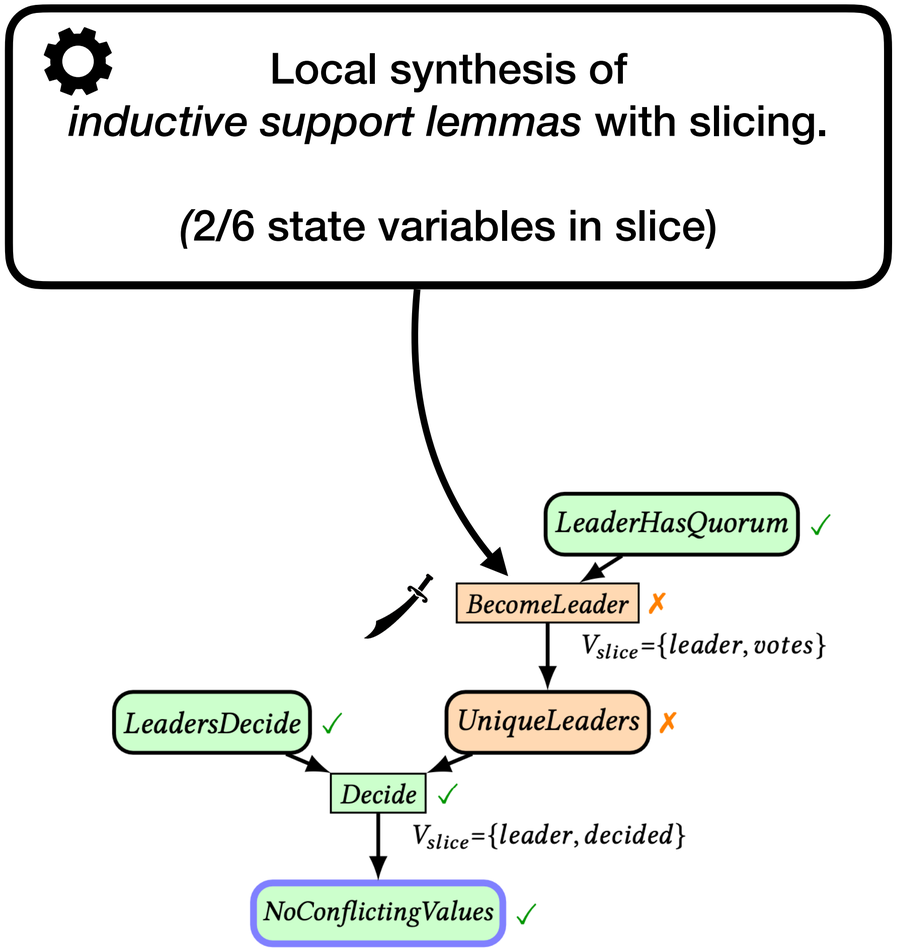
- Construct the inductive proof graph incrementally, backwards from safety property.



In-progress inductive proof graph.

Inductive Invariant Synthesis by Inductive Proof Slicing

- Construct the inductive proof graph incrementally, backwards from safety property.



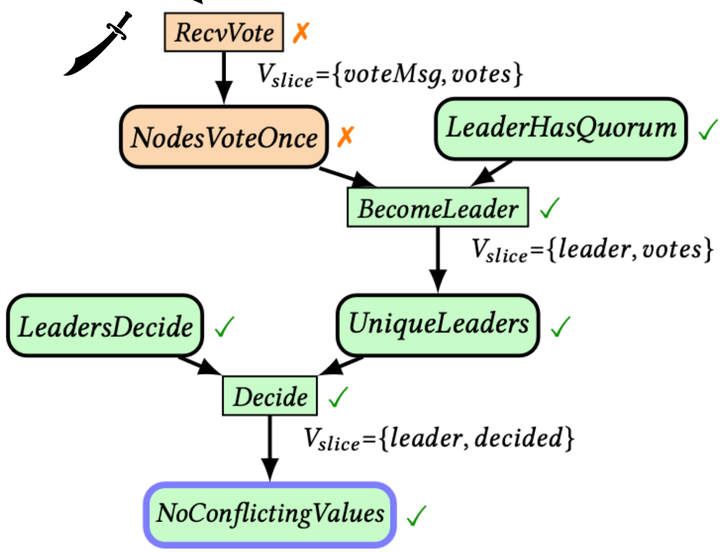
In-progress inductive proof graph.

Inductive Invariant Synthesis by Inductive Proof Slicing

- Construct the inductive proof graph incrementally, backwards from safety property.

Local synthesis of
inductive support lemmas with slicing.

(2/6 state variables in slice)



In-progress inductive proof graph.

Initial Evaluation

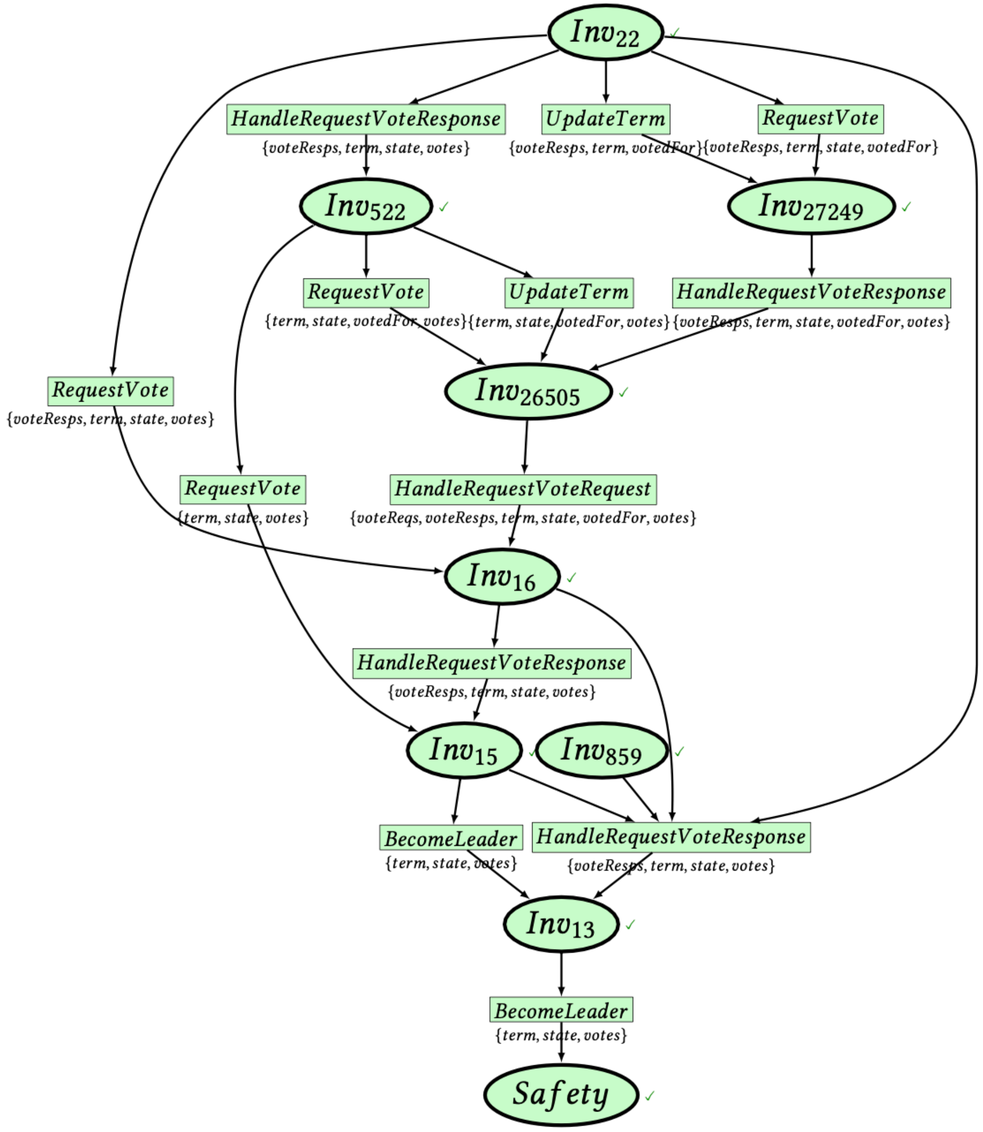
Scalability

AsyncRaft - large, asynchronous model of Raft consensus protocol.

~600 lines of TLA+ code in spec.

> 50 million reachable states in small finite model (N=3 processes).

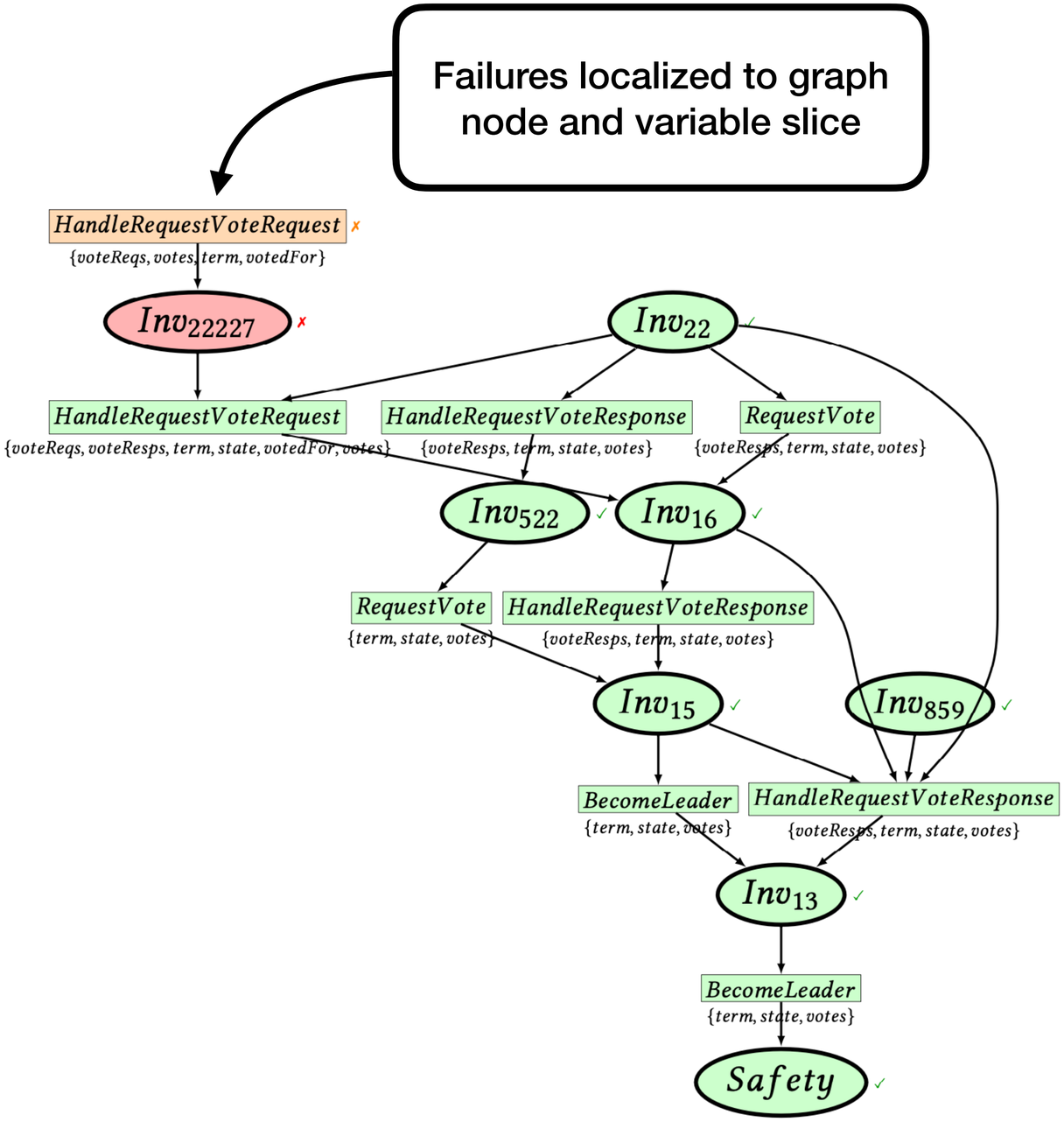
Synthesized inductive invariant for election safety property in ~3 hours.



Initial Evaluation

Interpretability

Incomplete grammar leading to global inference failure.



Thanks!

Work under submission, preprint available.