Verifiable Hierarchical Protocols with Network Invariants on Parametric Systems

Opeoluwa Matthews, Jesse Bingham, Daniel Sorin

http://people.duke.edu/~om26/
Problem Statement

- Goal: design and automated verification of hierarchical protocols

Safety property: $\forall L_1, \ldots, L_k. \text{Distinct}(L_1, \ldots, L_k) \Rightarrow P(L_1, \ldots, L_k)$
Problem Statement

• Parametric model checkers fall short
  • Suitable for flat protocols
  • Can’t handle asymmetry in hierarchical protocols

• Solution: Design specifically to fit automated techniques

• Formally specify class of transition systems – Neo
  • Require properties that enable automated safety verification
  • Key: Network invariants + parameterized verification
Illustration of our Approach

- **require** $L$ Network Invariant
  - All proper subtrees $P \leq L$
- Behavior along $c_1$ over-approximates $c_2$, $c_3$
- Preorder $\leq$ captures states and externally-visible behaviors of subhierarchy

R - Root node
I - Internal node
L - Leaf node
Neo Framework

• Neo formalized on I/O Automata (IOA) process theory

• Neo system is an IOA with specific properties for actions, composition, and executions

• 3 classes of IOA
  • Internal node
  • Leaf node
  • Root node

• Define 3 sets of actions
  • Upward actions – U
  • Downward actions – D
  • Peer-to-peer actions – P
**n-child k-peer Internal Node** $I$ is IOA that:

- Communicates with 1 parent, $n$ children, $k-1$ peers, with index $i$

\[
\begin{align*}
(p, i, 0) & \quad (p, i, 1) \\
(d, 0) & \quad (d, 1) \quad (d, 2) \quad (d, n-1) \\
(p, 0, i) & \quad (p, 1, i) \\
(u, 0) & \quad (u, 1) \quad (u, 2) \quad (u, n-1) \\
\end{align*}
\]

\[
\begin{align*}
(p, i, 2) & \quad (p, i, k-2) \\
(p, 2, i) & \quad (p, k-2, i) \\
\end{align*}
\]

\[
\begin{align*}
(u, i) & \\
(d, i) & \\
\end{align*}
\]

\[
\begin{align*}
\text{output actions} & \\
\text{input actions} & \\
\end{align*}
\]

\[
\begin{align*}
\text{u} \in U, \text{p} \in P, \text{d} \in D
\end{align*}
\]
**Leaf node** $L$ is 0-child, $k$-peer internal node:
- Communicates with 1 parent and $k$-1 peers, with index $i$

$$u \in U, p \in P, d \in D$$

**Output actions**
- $(u, i) \rightarrow L$
- $(p, i, 0) \rightarrow (p, i, 1)$
- $(p, i, 2) \rightarrow (p, i, k-2)$

**Input actions**
- $L \rightarrow (d, i)$
- $(p, 0, i) \rightarrow (p, 1, i)$
- $(p, 2, i) \rightarrow (p, k-2, i)$
n-child **Root Node** \( R \) is IOA that:
- Communicates with \( n \) children

\[
R \quad \xrightarrow{(d, 0)} \quad \xrightarrow{(d, 1)} \quad \cdots \quad \xrightarrow{(d, n-1)} \quad d \in D, \quad u \in U
\]

**output actions**

\[
R \quad \xleftarrow{(u, 0)} \quad \xleftarrow{(u, 1)} \quad \cdots \quad \xleftarrow{(u, n-1)} \quad \text{input actions}
\]
Defining Neo Systems

• $k$-peer Leaf $L$ is *Open Neo System*, communicates with $k-1$ peers
Defining Neo Systems

\[ \Omega = A \cdot \prod_{i=0}^{n-1} \Omega_i \]

\[ k\text{-peer internal node } A \rightarrow \]

\[ k\text{-peer Open Neo System} \]
Defining Neo Systems

\[ \Omega = A \cdot \prod_{i=0}^{n-1} \Omega_i \]

A is root node \( \rightarrow \) Closed Neo System
• *Network Invariants* captures behavior of subhierarchies (open Neo systems)
  • Require: Every open Neo system must implement leaf wrt \( \leq \)

• \( \leq \) captures summaries of states and executions
  • *Summary states*
  • *Summary functions*
  • *Summary sequences of executions*
• **Sum** is set of *summary states*, with special element *bad*

• Have $\text{sum}_*$ functions for every Neo system to capture summary state of each subhierarchy

• For leaf $L$, $\text{sum}_L : \text{states}(L) \rightarrow \text{Sum}$

• For each n-child root or internal node $A$, $\text{sum}_A : \text{states}(A) \times \text{Sum}^n \rightarrow \text{Sum}$

bad $\in \{s_0, \ldots, s_{n-1}\}$ implies $\text{sum}_A(s, s_0, \ldots, s_{n-1}) = \text{bad}$
Summarizing States – Neo systems

• For Neo system \( \Omega = A \cdot \prod_{i=0}^{n-1} \Omega_i \)

define \( \text{sum}_\Omega : \text{states}(\Omega) \rightarrow \text{Sum} \) as

\[
\text{sum}_\Omega(s_a, s_0, \ldots, s_{n-1}) =
\]

\[
\text{sum}_a(s_a, \text{sum}_{\Omega_0}(s_0), \ldots, \text{sum}_{\Omega_{n-1}}(s_{n-1}))
\]
Neo Safety

\[ s \in \text{states}(\Omega) \textbf{ safe if } sum_\Omega(s) \neq \text{bad} \]

\[ \Omega \textbf{ safe if all reachable states are safe} \]
Summarizing Executions

• Generate *summary sequence* of exec $e$ of $\Omega$ as follows:

$$e = s_0, \alpha_1, \ldots, \alpha_k, s_k$$

summarize states

$$\text{sum}_\Omega(s_0), \alpha_1, \ldots, \alpha_k, \text{sum}_\Omega(s_k)$$

Remove “silent” terms that don’t affect safety
Delete all $\alpha_i, \text{sum}_\Omega(s_i)$ such that

$$\alpha_i \in \text{int}(\Omega) \text{ and } \text{sum}_\Omega(s_i) = \text{sum}_\Omega(s_{i-1})$$
• Need preorder for network invariants

• Given 2 open Neo systems $\Omega_1, \Omega_2$

\[ \Omega_1 \preceq \Omega_2 \text{ implies for all executions } e_1 \text{ of } \Omega_1 \]

there exists execution $e_2$ \text{ of } $\Omega_2$

such that $\text{sum}(e_1) = \text{sum}(e_2)$
Theoretical Result

**Theorem 1.** (Every Neo system is safe.) Suppose that for each $n$-child internal or root node $A$, $\Omega_L = A \cdot \prod_{i=0}^{n-1} \phi_i(L)$ is safe. Furthermore, suppose that if $A$ is an internal node, then $\Omega_L \preceq L$. Then all Neo systems are safe.

**Antecedents:**

1. Every 1-level (all-leaf) open or closed neo system safe
2. Every 1-level (all-leaf) open neo system implements leaf

• If 1. and 2. can be performed in parametric model checker

**Implication:** Reduced 2-dimensional verification problem to 1 dimension
Case Study

- We design and verify hierarchical coherence protocol *NeoGerman*
- Modify (originally flat) German protocol into Neo hierarchy
- Coherence defined on predicates \{E, S, I\} on cache states
- 2 private caches in (E, E) or (E, S) prohibited
NeoGerman Protocol

• Root node is same as directory of German protocol
  • $\Omega_R$ is closed Neo system

• To get open Neo system $\Omega_I$, modify directory to be internal node (talk to parent)

• Internal node has state variable *Permissions*, captures summary of subhierarchy
NeoGerman Protocol Illustration

\[ \Omega_I \]

GetExclusive

\[ C_0 \quad C_1 \quad C_2 \quad \cdots \quad C_{n-1} \]

\[ l \quad S \quad S \quad S \]

Permissions=S
NeoGerman Protocol Illustration

\[ \Omega \]

GetExclusive

\[ D \]

Permissions=S

\[ C_0 \quad C_1 \quad C_2 \quad \ldots \quad C_{n-1} \]

\[ I \quad S \quad S \quad S \]
NeoGerman Protocol Illustration

\[ \Omega_I \]

\[ D \]

\[ C_0 \quad C_1 \quad C_2 \quad \ldots \quad C_{n-1} \]

\[ I \quad S \quad S \quad S \]

\[ \text{GrantExclusive} \]

\[ Permissions=E \]
NeoGerman Protocol Illustration

\[ \Omega \]

\[ I \]

\[ D \]

Permissions=E

\[ C_0, C_1, C_2, \ldots, C_{n-1} \]

\[ I, S, S, \ldots, S \]
NeoGerman Protocol Illustration

\[ \Omega \]

\[ D \]

Permissions = E

\[ C_0 \quad C_1 \quad C_2 \quad \ldots \quad C_{n-1} \]

\[ I \quad I \quad I \quad I \]
NeoGerman Protocol Illustration

\[ \Omega \]

\[ \begin{align*}
D & \quad \text{Permissions} = E \\
C_0 & \quad E \\
C_1 & \quad I \\
C_2 & \quad I \\
\cdots & \\
C_{n-1} & \quad I
\end{align*} \]
NeoGerman Summary Functions

- Preorder, safety defined w.r.t summary functions
- Need: if safety violated → function returns \textit{bad}

- Create ordering $<$ on $\text{Sum}$: $I < S < E < \text{bad}$
- 2 constraints on $\text{sum}_A$:

1) $\text{sum}_A(s_a, s_0, \ldots, s_{n-1}) = \text{bad}$ if $s_i = E$ and $s_j \neq I$

2) $s_i \leq \text{sum}_A(s_a, s_0, \ldots, s_{n-1})$

- Output of $\text{sum}_A$ always returns value of Permissions
Verification Methodology

• All verification done automated in Cubicle parametric model checker
  • SMT-based, backward reachability
  • Similar syntax to Murϕ, guard/action semantics
  • Clean, promising results, great support!

• Must prove antecedents of Theorem 1
  1. $\Omega_R$ and $\Omega_I$ safe – express in Cubicle
  2. $\Omega_I \preceq L$ (preorder) trickier
Preorder Proof

• Model both $\Omega_I$ and $L$ in same Cubicle program

• Force $\Omega_I$ and $L$ to transition in lockstep, starting with $\Omega_I$

• Have variables $O\_action$ and $L\_action$, represent IOA action, updated after each transition, internal actions updated to $\lambda$ (silent)

• One each transition, there needs to exist $L$ step that “matches” $\Omega_I$ step
  • To reveal witness step, conjunct expression to $L$ guards, forcing $L$ take “right” step w.r.t $\Omega_I$ step.
  • Note: conjunction can only restrict $L$ behavior
After each $\Omega_I$ step, Cubicle checks:

- There exists $L$ action that can fire
  - Cubicle safety prop: Disjunction of all $L$ guards is true

After each pair of $\Omega_I$ and $L$ steps, Cubicle checks:

- $O\_action=L\_action$, summary state outputs match
What Safety Properties can Neo Verify?

• Define class of FOL formulas we can verify are invariant

Given set $LP = \{p_1, \ldots, p_m\}$ of predicates on leaf states and proposition logic formula $P(L_1, \ldots, L_k)$ over atoms of form $p_j(L_i)$

• We can verify all safety properties of the form:

$\forall L_1, \ldots, L_k. Distinct(L_1, \ldots, L_k) \Rightarrow P(L_1, \ldots, L_k)$

• E.g., $LP=\{E,S,I\}$  $\forall L_1, L_2. Distinct(L_1, L_2) \Rightarrow (E(L_1) \Rightarrow I(L_2))$

• We provide summary function guaranteed to verify all such safety properties
Future Work

- Industrial-strength hierarchical coherence protocol
  - Request forwarding
  - MESI coherence permissions
  - Support for unordered networks
- Distributed lock management
  - Richer permissions (NL, CR, CW, PR, PW, EX)
- Dynamic power management
  - Natural hierarchy in datacenters
Conclusions

• Neo framework enables design and automated verification of hierarchical protocols safe for arbitrary configurations

• Case study: Design and verify hierarchical coherence protocol
  • Correct for arbitrary size, depth, branching degrees per node
  • Proof completely automated in parametric model checker

• Prove observational preorder in parametric setting

http://people.duke.edu/~om26/