Actual Causality in Reactive Systems

Hadar Frenkel

February 2023
Explaining Hyperproperty Violations

Norine Coenen¹, Raimund Dachselt², Bernd Finkbeiner¹, Hadar Frenkel¹, Christopher Hahn¹, Tom Horak², Niklas Metzger¹, and Julian Siber¹

¹ CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
² Interactive Media Lab, Technische Universität Dresden, Dresden, Germany
³ elevat GmbH & Co. KG, Dresden, Germany
dachselt@acm.org
tom.horak@elevat.de

Abstract. Hyperproperties relate multiple computation traces to each other. Model checkers for hyperproperties thus return, in case a system model violates the specification, a set of traces as a counterexample. Fixing the erroneous relations between traces in the system that led to the counterexample is a difficult manual effort that highly benefits from additional explanations. In this paper, we present an explanation method for counterexamples to hyperproperties described in the specification logic HyperLTL. We extend Halpern and Pearl’s definition of actual causality to sets of traces witnessing the violation of a HyperLTL formula, which allows us to identify the events that caused the violation. We report on the implementation of our method and show that it significantly improves on previous approaches for analyzing counterexamples returned by HyperLTL model checkers.

Temporal Causality in Reactive Systems

Norine Coenen¹, Bernd Finkbeiner¹, Hadar Frenkel¹, Christopher Hahn², Niklas Metzger¹, and Julian Siber¹

¹ CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
² Stanford University, Stanford, USA

Abstract. Counterfactual reasoning is an approach to infer what causes an observed effect by analyzing the hypothetical scenarios where a suspected cause is not present. The seminal works of Halpern and Pearl have provided a workable definition of counterfactual causality for finite settings. In this paper, we propose an approach to check causality that is tailored to reactive systems, i.e., systems that interact with their environment over a possibly infinite duration. We define causes and effects as trace properties which characterize the input and observed output behavior, respectively. We then instantiate our definitions for ω-regular properties and give automata-based constructions for our approach. Checking that an ω-regular property qualifies as a cause can then be encoded as a hyperproperty model-checking problem.
Model Checking

WHY?
Model Checking

Reactive systems

Inputs trigger outputs

Infinite computations

{g₁} → r₁ → ¬ r₁ ∧ r₂ → {g₂}

∅ → r₁ ∧ ¬ r₂ → ∅
Model Checking

Is “always $r_1$” the cause for “always not $g_2$”? 

Causes over input sequences
Analyse the system dynamics

“eventually $g_2$”
Causality as a Hyperproperty

Is “always $r_1$” the cause for “always not $g_2$”? Compare $\pi$ with the counterfactual trace $\pi'$. 

$\forall \pi \exists \pi' \phi$
Actual Causality in Reactive Systems

**Outputfiles**

<table>
<thead>
<tr>
<th>Generated Alger</th>
<th>Generated Dot</th>
<th>Counter Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>in_00=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in_100=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 I: (And(Or(Neg(light0))(Neg(light1)))(Or(light0)(light1)))...284#0=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 I: remember_state@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>light_00=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 state[0]_00=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 state[1]_00=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 light_100=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 state[0]_100=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 state[1]_100=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 sink@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 init@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13 entered_lasso@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14 L: (And(Or(Neg(light0))(Neg(light1)))(Or(light0)(light1)))...284#0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 I: copy@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 I: copy@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17 I: copy@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18 I: copy@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19 I: copy@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 I: copy@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21 I: copy@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22 I: copy@0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23 _M0: (And(Or(Neg(light0))(Neg(light1)))(Or(light0)(light1)))...284#0=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24 in_00=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25 in_10=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26 I: (And(Or(Neg(light0))(Neg(light1)))(Or(light0)(light1)))...284#1=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27 I: remember_state@1=0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28 I: copy@1=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29 state[0]_00=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30 state[1]_00=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>31 light_10=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32 light_10=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>33 light_10=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>34 light_10=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>35 light_10=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>36 light_10=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>37 light_10=1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. \(\{\langle p, \pi, 0\rangle\}\)
   - Specific events on the trace that cause the violation

2. “infinitely often p”
   - Trace properties

**MCHyper**

- Explainability — analysis of the counterexample
- Applicability — repair
Causality in reactive systems

Causes as sets of events

Causes as temporal properties

Causality as a Hyperproperty
Hyperproperties

• Extend trace properties (e.g., in LTL) to system properties

• Reason about sets of traces

\[ \forall \pi \exists \pi' \varphi \]

Linear Temporal Logic — LTL

\[ \square p \rightarrow p \text{ holds at every timepoint} \]

\[ \Diamond p \rightarrow p \text{ eventually holds} \]

\[ \Diamond p \rightarrow p \text{ holds at the next timepoint} \]
Hyperproperties

- Extend trace properties (e.g., in LTL) to system properties
- Reason about sets of traces
- Observational determinism:
  \[
  \forall \pi \forall \pi' \Box (li_\pi \leftrightarrow li_{\pi'}) \rightarrow \Box (lo_\pi \leftrightarrow lo_{\pi'})
  \]

Hyperproperties. Clarkson and Schneider. (CSF 2008).
Hyperproperties

• Extend trace properties (e.g., in LTL) to system properties

• Reason about sets of traces

Hyperproperties. Clarkson and Schneider. (CSF 2008).
Explaining Hyperproperty Violations

\[ \forall \pi \forall \pi' \quad \Box (l_{i\pi} \leftrightarrow l_{i\pi'}) \rightarrow \Box (l_{o\pi} \leftrightarrow l_{o\pi'}) \]

WHY?

\[ \emptyset \{lo\} \]

\[ \neg hi \]

\[ \pi \]

\[ \pi' \]

\[ \omega \]

\[ \omega \]
Actual Causality
Actual Causality

Hyperproperties
Relate multiple system executions

Actual world

Counterfactual world

Actual Causality

Actual Causality

Actual world

Counterfactual world

Billy's Ball breaks the bottle!

Actual Causality

Contingencies

Actual world

Counterfactual world + contingency

preemption of causes

Actual Causality

AC1: the cause appears in the actual world

AC2: for every counterfactual world there exists a contingency where effect does not hold

AC3: this is a minimal cause

Actual Causality

SAT: the cause appears in the actual world

CF: for every counterfactual world there exists a contingency where effect does not hold

MIN: this is a minimal cause

Causality in reactive systems

Causes as sets of events


Causality as a Hyperproperty

Causality Checking for Complex System Models. Leitner-Fischer, Leue. (VMCAI 2013)
Actual Causality for Hyperproperties

SAT

CF: for every counterfactual world there exists a contingency where effect does not hold

MIN
Actual Causality for Hyperproperties

- Effect: a violation of a **Hyperproperty** $\psi$
- Actual World: a set $\Gamma$ of counterexample traces
- Cause: set of events on the set of traces

∀* prefix:

$\psi = \forall \pi_1 \forall \pi_2 \exists \pi'_1 \exists \pi'_2 . \varphi$

$\neg \psi = \exists \pi_1 \exists \pi_2 \forall \pi'_1 \forall \pi'_2 . \neg \varphi$

Lasso-shaped
Explaining Hyperproperty Violations

∀ π ∀ π' □ (li_π ↔ li_π') → □ (lo_π ↔ lo_π)

 WHY? 
Explaining Hyperproperty Violations

CF: \( \forall \) counterfactual \( \exists \) contingency s.t. \( \varphi \) holds

\[ \forall \pi \forall \pi' \Box (li_\pi \leftrightarrow li_{\pi'}) \rightarrow \Box (lo_\pi \leftrightarrow lo_{\pi'}) \]
Explaining Hyperproperty Violations

\[ \forall \pi \forall \pi' \Box (li_\pi \leftrightarrow li_{\pi'}) \rightarrow \Box (lo_\pi \leftrightarrow lo_{\pi'}) \]

\[ C = \{ \langle hi, 0, \pi' \rangle \} \]

\[ \Gamma = (\pi, \pi') \]

CF: \forall \text{ counterfactual } \exists \text{ contingency s.t. } \varphi \text{ holds}
Explaining Hyperproperty Violations

\[ \forall \pi \forall \pi' (\Box (li_\pi \leftrightarrow li_{\pi'}) \rightarrow \Box (lo_\pi \leftrightarrow lo_{\pi'})) \]

\[ C = \{ \langle hi, 0, \pi' \rangle \} \]

\[ \Gamma = (\pi, \pi') \]

CF: \( \forall \) counterfactual \( \exists \) contingency s.t. \( \varphi \) holds

Flip all events in \( C \)

Setting back to values of the original world

intervene(\( \Gamma, C, \{ \langle lo, 2, \pi' \rangle \} \))
Computing Contingencies

Counterfactual automaton: additional inputs $[c]$ to set a contingency

an input $c_o$ for each output $o$
Actual Causality for Hyperproperties

Find $C$ such that

- **SAT:** $\Gamma \not\models C$
- **CF:** $\forall$ counterfactual $\exists$ contingency s.t. $\varphi$ holds
- **MIN:** no subset of $C$ satisfies SAT & CF

Finding a cause as a hyperproperty

$\exists \pi_1 \exists \pi_2 \cdot \forall \pi_1' \forall \pi_2' \cdot \psi_{cause}$

Events on $\pi_1, \pi_2$ correspond to the cause
$\pi_1', \pi_2'$ represent other possible (not minimal) causes

Algorithms for Model Checking HyperLTL and HyperCTL*. Finkbeiner, Rabe, Sánchez. (CAV 2015)
Computing Actual Causes

largest candidate cause $C$ — SAT dependencies

\[ \forall \pi_1, \ldots, \forall \pi_n \cdot \varphi \]

\[ \Gamma = \pi_1, \ldots, \pi_n \]
Computing Actual Causes

largest candidate cause $C$ — SAT dependencies

SAT: $\Gamma \models C$

CF: $\forall$ counterfactual $\exists$ contingency s.t. $\varphi$ holds

MIN: no subset of $C$ satisfies SAT & CF
Computing Actual Causes

largest candidate cause $C$ — SAT dependencies

SAT: $\Gamma \models C$

CF: $\forall$ counterfactual $\exists$ contingency s.t. $\varphi$ holds

MIN: no subset of $C$ satisfies SAT & CF
## Experiments

| Instance                        | $|\Gamma|$ | $|\varphi|$ | $(C)$ | time (ms) |
|---------------------------------|---------|---------|------|---------|
| Running example (paper)         | 10      | 9       | 2    | 55      |
| Security in & out               | 35      | 19      | 8    | 798     |
| Drone example 1                 | 24      | 19      | 5    | 367     |
| Drone example 2                 | 18      | 36      | 3    | 256     |
| Asymmetric arbiter '19          | 28      | 35      | 10   | 490     |
| Asymmetric arbiter              | 72      | 35      | 24   | 1480    |

\[
\{ \langle hi, 0, \pi \rangle \} \\
\{ \langle \neg hi, 0, \pi \rangle \} 
\]
Causality in reactive systems

Causes as sets of events

Causes as temporal properties

Causality as a Hyperproperty

Hyperproperties

Halpern & Pearl Causality
Causes as Trace Properties

- Effect: a violation of an $\omega$-regular property $\psi$
- Actual World: a counterexample trace
- Cause: an $\omega$-regular property

Quantified Propositional Temporal Logic — QPTL

LTL + quantification over propositions

$$\exists q . q \land \Box (q \leftrightarrow \Diamond \neg q) \land \Box (q \to a)$$ — “a holds at every odd position”
Causes as Trace Properties

\begin{align*}
\pi \left( \left( \begin{array}{c}
{g_1} \\
\varnothing \\
{g_2}
\end{array} \right) \right) \omega & \nleq \Diamond g_2
\end{align*}
Causes as Trace Properties

\[ C = r_1 \land \Diamond r_1 \]
\[ \neg C = \neg r_1 \lor \Diamond \neg r_1 \]

Closest input sequences s.t. \( C \) doesn’t hold

\[ \forall \, \exists \, \lozenge g_2 \text{ holds} \]

\[ \pi \left( \begin{array}{c} r_1 \ r_2 \\ r_1 \ r_2 \\ r_1 \ r_2 \\ g_1 \end{array} \right)^{\omega} \not\models \lozenge g_2 \]
Causes as Trace Properties

Closest input sequences
\[ C = \square \Diamond \neg r_1 \]
\[ \neg C = \Diamond \square \neg r_1 \]

Compare traces that have the same rejection structure

\[ \text{CF: } \forall \text{ counterfactual } \exists \text{ contingency s.t. } \Diamond g_2 \text{ holds} \]

\[ \top \neq \Diamond g_2 \]
Causes as Trace Properties

Compare traces that have the same rejection structure

HyperQPTL formula

$\psi_{struct}(\pi_1, \pi_2) : \pi_1, \pi_2$ satisfy all subformulas of $C$ at the same positions

Closest input sequences s.t. $C$ doesn’t hold
Causes as Trace Properties

CF: $\forall$ counterfactual $\exists$ contingency s.t. $\Diamond g_2$ holds

Counterfactual automaton additional inputs $[c_1, c_2]$ set a contingency
Causes as Trace Properties

MIN: There is no $C'$ such that $C' \rightarrow C$
and $C' \models \text{SAT} & \text{CF}$

HyperQPTL formula

No lasso-shaped trace can be removed from $C$

a trace that does not satisfy the effect, or does not contribute for counterfactual traces
Causes as Trace Properties

Given a candidate cause $C$, verify:

SAT: $\pi \not\models C$

CF: $\forall$ counterfactual $\exists$ contingency s.t. $\varphi$ holds

MIN: There is no $C'$ such that $C' \rightarrow C$ and $C' \not\models SAT$ & $CF$

$\Rightarrow C$ is a cause of $\varphi$ on $\pi$

HyperQPTL

Counterfactuals: traces with the same rejection structure
Contingencies: using the counterfactual automaton

Decidable via HyperQPTL model-checking!

Unfair Arbiter

Is □ \( r_1 \) the cause for □ \( \neg g_2 \)?
Unfair Arbiter

\[ \exists q . q \land \Box (q \leftrightarrow \Diamond \neg q) \land \Box (q \rightarrow r_1) \]\n
\( r_1 \) holds at every odd position

is a cause for \( \Box \neg g_2 \) on \( \pi \)
Causality in reactive systems

Causes as sets of events

Verifying a cause via HyperQPTL model checking

Causes as temporal properties

Formalism for expressing HP causality for reactive systems

Finding a cause via HyperLTL model checking

Algorithm for finding all causes
Causality in reactive systems
Causes as sets of events

Thank you!
Questions?