Runtime Verification of Temporal Properties over Out-of-Order Streams

Felix Klaedtke
NEC Labs Europe, Heidelberg

Joint work with David Basin (ETHZ) and Eugen Zalinescu (TUM)
(1) Leverage local advantages at five labs to create No.1/Only 1 technologies, create new business through collaboration between labs

(2) Reinforce global open innovation to build solutions

NEC's Global R&D Activities

- **NEC Labs. Europe**
  - Core technology development through standardization and EU projects and R&D marketing

- **NEC Labs. China**
  - Develop new solutions for a sizable market

- **NEC Labs. America, Inc**
  - Develop core technologies by leveraging presence in the country of cutting-edge technology and business

- **NEC Labs. Singapore**
  - Use core technologies from other labs for collaboration and demo experiments with local government and customers
  - Established in 2013

- **Central Research Laboratories (Japan)**
  - The control tower for research
  - Develop new solutions in collaboration with customers/divisions
  - Points of university-industry collaboration and verification
~100 leading researchers from all over Europe and world-wide in Heidelberg, and London/S.Ruislip (NEC E HQ)

Research addressing European and world-wide technology and business trends on

- **Technology Platforms:** 5G, SDN/VNF, security, data science, IoT
- **Solutions:** smart transport, smart cities, public safety, industrial ctrl.

Close **collaboration with**

- top European universities and research institutes
- major industry in Europe, e.g., network operators, automotive, utilities, ...

**NEC Laboratories Europe**

- http: www.neclab.eu
- email: quittek@neclab.eu
Runtime Verification in a Nutshell

\[ \Box p \rightarrow \Diamond_{<3} q \]
Talk Overview

1. traces

2. specification language

3. monitoring

1. system model

4. experiments

\( \square p \rightarrow \diamond_{<3} q \)
System Models

- Synchronous product
- State machines
- LTL to automata translations
  ...

- Distributed components
- Asynchronous communication
- Unreliable channels
  ...

© NEC Corporation 2017
System Setup and Assumptions

- **Fixed** number of system components
- Components perform **actions**
- **Single** monitor observes actions
- One-way **channels** to monitor

Channels to monitor are **unreliable**
- Nonuniform delivery delays and message loss
- A message uniquely describes performed action
- No bogus messages
- No tampering with messages

Components **timestamp** messages to monitor
- Timestamps are **precise**
On the Timestamp Assumption

**Benefit:** Actions can be *linearly* ordered
- Logical clocks would only allow us to order the actions partially
- Hybrid clocks exist and provide an improvement over wall clocks

**Synchronized clocks do not exist in distributed systems**
- Clock drifts
- Clock skews (when assuming a global clock)
- Time protocols like NTP can achieve a precision in the millisecond range

**Verdicts reported by the monitor are based on its observations**
- Observations (timestamps) may contain an error
- Whether this imprecision is acceptable is application dependent
- Properties can to some extend take this imprecision into account
Observations (informal)

The system behavior is an infinite sequence of timestamped actions.
At most the actions up to the current time are known to the monitor.
Because of delivery delays the prefix might contain gaps.
Gaps may get resolved later.
Observations

Observations are finite words with letters of the form \((I, \sigma)\)

- \(I\) is a nonempty interval
- \(\sigma\) is a partial function describing an action

**Base case**
The word \(([0, \infty), [\ ]])\) of length 1 is an observation

**Step case**
Replace a letter with nonsingleton interval \(I\) in an observation:

\[
\begin{align*}
(I, [\ ]) & \mapsto (I \cap [0, \tau), [\ ]) ([\tau], \sigma) (I \cap (\tau, \infty), [\ ]) \\
([0, b), [\ ]) & \mapsto ([0], \sigma) ((0, b), [\ ]) \\
(I, [\ ]) & \mapsto \varepsilon \text{ if } |I| < \infty
\end{align*}
\]

**Intuition:**
- An observation \((I_1, \sigma_1) \ldots (I_n, \sigma_n)\) corresponds to the monitor’s knowledge about the system behavior
- Whenever the monitor receives a message, the monitor refines its knowledge

\[
([0, \infty), [\ ]) = w_0 \subseteq w_1 \subseteq w_2 \subseteq \ldots \subseteq t \in T(S)
\]
Metric Temporal Logic over Observations

Syntax

\[ \phi ::= t | p(x_1, ..., x_n) | \neg \phi | \phi \lor \phi | \phi S_I \phi | \phi U_I \phi | \downarrow^r x. \phi \]

Kleene logic (f, t, and \( \perp \) “unknown” with \( \perp < f \) and \( \perp < t \))

<table>
<thead>
<tr>
<th>( \neg )</th>
<th>( \lor )</th>
<th>( \land )</th>
<th>( \rightarrow )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f )</td>
<td>( f )</td>
<td>( f )</td>
<td>( f )</td>
</tr>
<tr>
<td>( t )</td>
<td>( f )</td>
<td>( t )</td>
<td>( t )</td>
</tr>
<tr>
<td>( \perp )</td>
<td>( \perp )</td>
<td>( \perp )</td>
<td>( \perp )</td>
</tr>
</tbody>
</table>

Semantics (nontemporal connectives)

\[ \llbracket w, i, \nu \vDash t \rrbracket := t \]

\[ \llbracket w, i, \nu \vDash p(\bar{x}) \rrbracket := \begin{cases} f & \nu(\bar{x}) \text{ def. and } \nu(\bar{x}) \not\in \sigma_i(p) \\ t & \nu(\bar{x}) \text{ def. and } \nu(\bar{x}) \in \sigma_i(p) \\ \perp & \text{otherwise} \end{cases} \]

\[ \llbracket w, i, \nu \vDash \neg \phi \rrbracket := \neg \llbracket w, i, \nu \vDash \phi \rrbracket \]

\[ \llbracket w, i, \nu \vDash \phi \lor \psi \rrbracket := \llbracket w, i, \nu \vDash \phi \rrbracket \lor \llbracket w, i, \nu \vDash \psi \rrbracket \]

\[ \llbracket w, i, \nu \vDash \downarrow^r x. \phi \rrbracket := \llbracket w, i, \nu[x \mapsto \sigma_i(r)] \vDash \phi \rrbracket \]
Temporal Connectives

Recall:

\[ p \cup_{I} q \]

Caveat:

- Letters with nonsingleton intervals might not contain any time points
- Anchor: existential statement about a time point
- Continuation: universal statement over time points

Adjust truth values at positions with nonsingleton intervals:

\[
[w, i, \nu \models \phi \cup_{I} \psi] := \bigvee_{j \geq i} (tp^w(j) \land tc^w(I, i, j) \land [w, j, \nu \models \psi] \land \\
\bigwedge_{i \leq k < j} (tp^w(k) \rightarrow [w, k, \nu \models \phi]))
\]
Properties

Artifact:
- $[w, i, v \models \phi] \neq \bot$ possible for $|I_i| > 1$
- Downgrade truth value at top level: $[w, \tau, v \models \phi] := \begin{cases} [w, i, v \models \phi] & \text{if } I_i = \{\tau\} \\ \bot & \text{otherwise} \end{cases}$

Monotonicity:
If $u \sqsubseteq v$ and $\mu \sqsubseteq \eta$ then $[u, \tau, \mu \models \phi] \preceq [v, \tau, \eta \models \phi]$

Word problem: PSPACE-complete
- Hardness: by reduction from QBF satisfiability problem
- Membership: use inductive definition of $[]$
- In PTIME for propositional formulas: use dynamic programming

Corresponding problem in the Boolean setting:
- PSPACE-complete for LTL (over infinite words)
- Undecidable for MTL (over timed words)
Monitor Correctness Requirements

- Input: \( w_0, w_1, w_2, \ldots \) with \( w_0 \subseteq w_1 \subseteq w_2 \subseteq \cdots \)
- Output: \( V_0, V_1, V_2, \ldots \) with \( V_i = \{ (\tau_1, b_1), \ldots, (\tau_n, b_n) \} \)

- Observational soundness:
  If \((\tau, b) \in V_i\) then \([w_i, \tau, \nu \models \phi] = b\)

- Observational completeness:
  If \([w_i, \tau, \nu \models \phi] \neq \bot\) then \((\tau, [w_i, \tau, \nu \models \phi]) \in \bigcup_{j \leq i} V_j\)

- Observationally sound and complete monitors exist!
- Observationally sound and complete monitors output verdicts promptly
Algorithmic Realization: Main Loop

1. Init(\(\phi\))
2. loop {
3.     event ← Receive(in)
4.     SplitInterval(event)
5.     PropagateDataValues(event)
6.     PropagateTruthValues(event)
7. }
Algorithmic Realization: State Updates (1)

$p \rightarrow \diamond_{(0,2)} q$

$p$

$q$

$\diamond_{(0,2)} q$

$[0, \infty)$

$[0, 1)$

$\{1\}$

$(1, \infty)$
Algorithmic Realization: State Updates (2)

\[ p \rightarrow \otimes_{(0,2)} q \]

\[ (1, t) \]

\[ t \]

\[ 1 \]

\[ t \]

\[ 2 \]

\[ q \]

\[ p \]

\[ \Diamond_{(0,2)} q \]

\[ \{1\} \]

\[ (1, \infty) \]

\[ \{2\} \]

\[ (2, \infty) \]
Algorithmic Realization: State Updates (3)

\[ \downarrow x.p \rightarrow \Diamond_{(0,2)} q(x) \]

\[ p \rightarrow \Diamond_{(0,2)} q(x) \]

\[ \Diamond_{(0,2)} q(x) \]

\[ q(x) \]

\[ [0, \infty) \]

\[ [0, 1) \]

\[ \{1\} \]

\[ (1, \infty) \]
Experimental Evaluation

Prototype in the programming language Go

- Command-line tool
- Reads streams either from a log file or a UDP socket
- Current version does not extensively use goroutines and channels

Setup:

- Computer with 2.8 GHz CPU and 8 GB RAM
- Synthetic logs over 60 seconds with different event rates (events per second)
- Scrambled logs by Gaussian distribution with different standard deviations $\sigma > 0$
- Test formulas:

\[
\square \downarrow_{cid} c \downarrow_{tid} t \downarrow_{\text{sum}} a \cdot \text{trans}(c, t, a) \land a > 2000 \rightarrow \Diamond_{[0,3]} \text{report}(t) \tag{P1}
\]
\[
\square \downarrow_{cid} c \downarrow_{tid} t \downarrow_{\text{sum}} a \cdot \text{trans}(c, t, a) \land a > 2000 \rightarrow \Box_{(0,3)} \downarrow_{tid'} t' \downarrow_{\text{sum}} a' \cdot \text{trans}(c, t', a') \rightarrow a' \leq 2000 \tag{P2}
\]
\[
\square \downarrow_{cid} c \downarrow_{tid} t \downarrow_{\text{sum}} a \cdot \text{trans}(c, t, a) \land a > 2000 \rightarrow (\downarrow_{tid'} t' \downarrow_{\text{sum}} a' \cdot \text{trans}(c, t', a') \rightarrow t = t') \text{W report}(t) \tag{P3}
\]
\[
\square \downarrow_{cid} c \downarrow_{tid} t \downarrow_{\text{sum}} a \cdot \text{trans}(c, t, a) \land a > 2000 \rightarrow \Box_{[0,6]} \downarrow_{tid'} t' \downarrow_{\text{sum}} a' \cdot \text{trans}(c, t', a') \rightarrow \Diamond_{[0,3]} \text{report}(t') \tag{P4}
\]

... and “propositional counterparts”
Experimental Results

Propositional setting

in-order delivery

out-of-order delivery (event rate 1000)

Half-order setting

in-order delivery

out-of-order delivery (event rate 100)

Comparison with MONPOLY (in-order delivery)

- Propositional setting: MONPOLY is faster by a factor 3-5
- Half-order setting: MONPOLY is several magnitudes faster
Conclusions and Future Work

### Novel monitoring approach
- It is based on a three-valued semantics for MTL
- It accounts for message loss and delays (no buffering)
- It provides soundness and completeness guarantees

### Preliminary experimental results
- For propositional streams: good performance
- For data streams: acceptable performance (depending on the application)

### Ongoing and future work
- Improve algorithms and optimize prototype
- Undertake case studies (runtime verification of SDN network components)
References

D. Basin, F. Klaedtke, and E. Zalinescu
*Failure-aware runtime verification of distributed systems*
FSTTCS 2015

D. Basin, F. Klaedtke, and E. Zalinescu
*Runtime verification of temporal properties over out-of-order data streams*
CAV 2017 (to appear)
NEC brings together and integrates technology and expertise to create the ICT-enabled society of tomorrow. We collaborate closely with partners and customers around the world, orchestrating each project to ensure all its parts are fine-tuned to local needs. Every day, our innovative solutions for society contribute to greater safety, security, efficiency and equality, and enable people to live brighter lives.
Orchestrating a brighter world

NEC