Adventures in Monitorability

Antonis Achilleos[1]

joint work with:

Luca Aceto[1,2] Adrian Francalanza[3] Anna Ingólfsdóttir[1]
Karoliina Lehtinen[4,5]

1: Reykjavik University

2: Gran Sasso Science Institute, L'Aquila

3: ICT, University of Malta

4: Christian-Albrechts University of Kiel

5: University of Liverpool

NYCAC 2018 16 November 2018

Formal Verification

- Question: is your system behaving correctly?

Formal Verification

- Question: is your system behaving correctly?
- Multiple verification techniques: Model-Checking, Theorem-Proving, Testing,...

Formal Verification

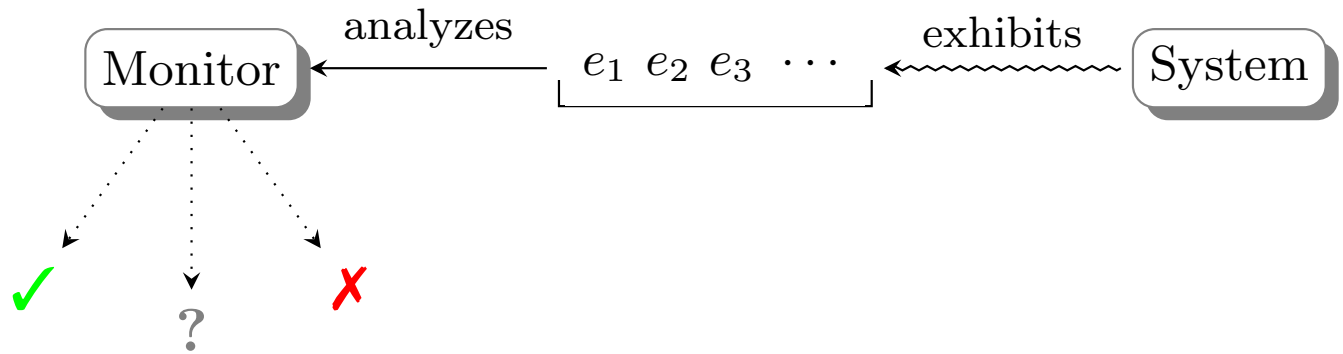
- Question: is your system behaving correctly?
- Multiple verification techniques: Model-Checking, Theorem-Proving, Testing,...
- Issues: systems become (even) larger and more complicated, unexpected environments, opaque components

Formal Verification

- Question: is your system behaving correctly?
- Multiple verification techniques: Model-Checking, Theorem-Proving, Testing,...
- Issues: systems become (even) larger and more complicated, unexpected environments, opaque components
- A post-deployment technique: Runtime Verification

Runtime Verification

- Runtime Verification uses monitors to detect at runtime whether a certain system satisfies/violates a specification.
- A monitor runs together with a process and it observes the events the process generates.
- When it detects a certain kind of behavior, it can reach a verdict (`yes`, `no`, or `end`).



Goals

Automatic monitor synthesis from specifications

Plausible monitorability guarantees for classes of properties

To determine the limits of monitorability

Goals

Automatic monitor synthesis from specifications

Plausible monitorability guarantees for classes of properties

To determine the limits of monitorability

A choice to make:

properties of
- processes
- infinite traces
- finite or infinite traces

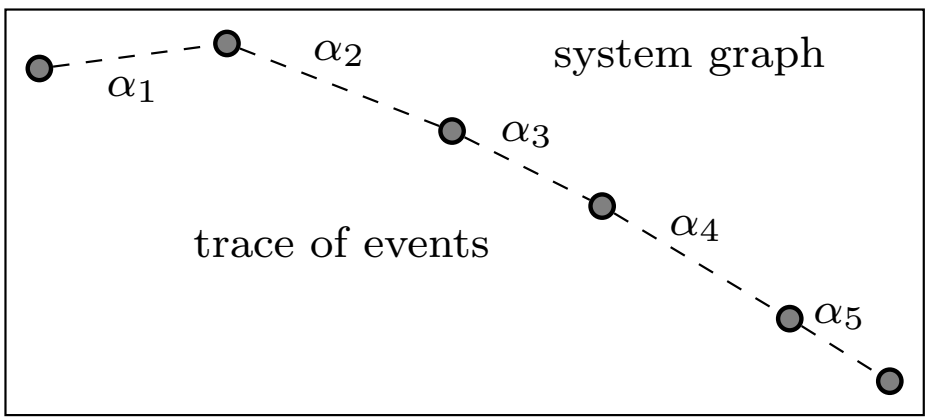
Two Kinds of Models – Two Kinds of Properties

Processes, Infinite Traces

$$\langle P, \text{Act}, \rightarrow \rangle$$

processes, actions, transitions

$$p \xrightarrow{\alpha_1} q \xrightarrow{\alpha_2} \cdots$$



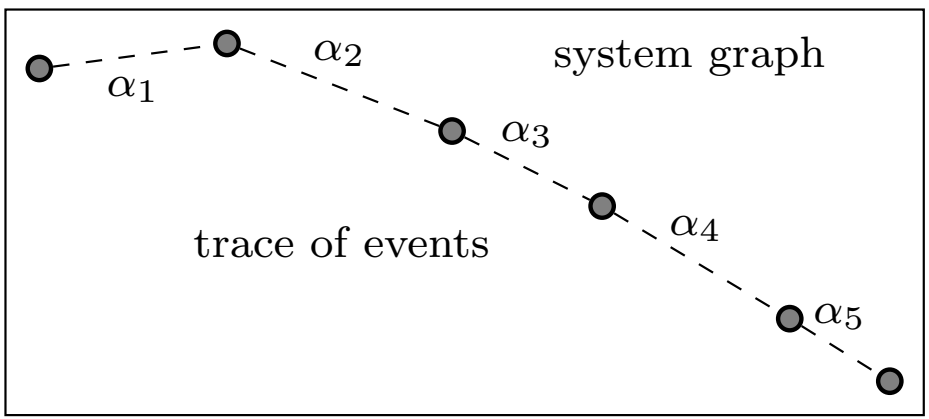
Two Kinds of Models – Two Kinds of Properties

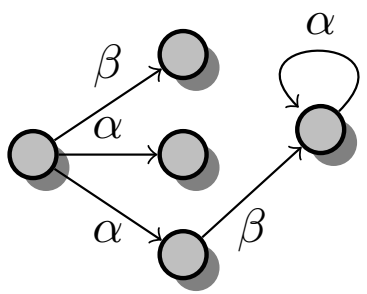
Processes, Infinite Traces

$$\langle P, \textsc{Act}, \rightarrow \rangle$$

processes, actions, transitions

$$p \xrightarrow{\alpha_1} q \xrightarrow{\alpha_2} \cdots$$





$$\alpha_1\alpha_2\alpha_3\cdots \in \textsc{Act}^\omega$$

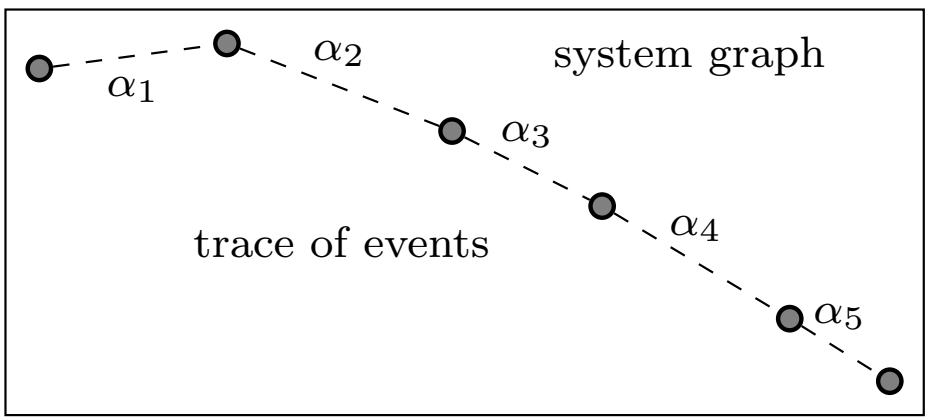
Two Kinds of Models – Two Kinds of Properties

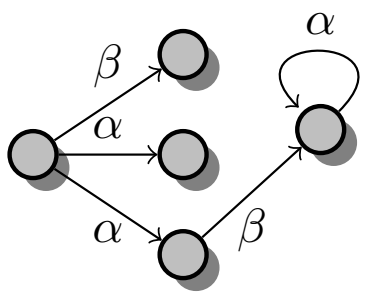
Processes, Infinite Traces

$$\langle P, \textsc{Act}, \rightarrow \rangle$$

processes, actions, transitions

$$p \xrightarrow{\alpha_1} q \xrightarrow{\alpha_2} \cdots$$





$$\alpha_1\alpha_2\alpha_3\cdots \in \textsc{Act}^\omega$$

Moral of the talk: The choice of model matters!

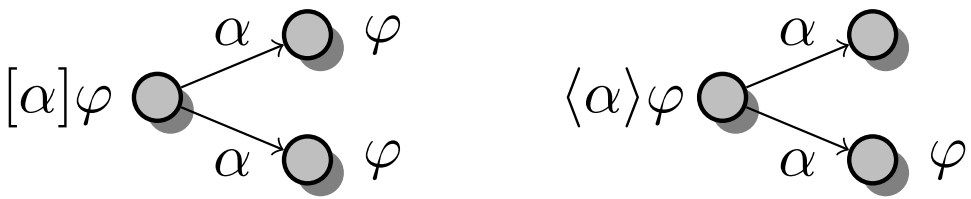
The Language

$$\varphi, \psi \in \mu\text{HML} ::= \texttt{tt} \mid \langle\alpha\rangle\varphi \mid \varphi \vee \psi \mid \min X.\varphi \mid X$$
$$\mid \texttt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \max X.\varphi$$

The Language

$$\varphi, \psi \in \mu\text{HML} ::= \texttt{tt} \mid \langle\alpha\rangle\varphi \mid \varphi \vee \psi \mid \min X.\varphi \mid X$$
$$\mid \texttt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \max X.\varphi$$

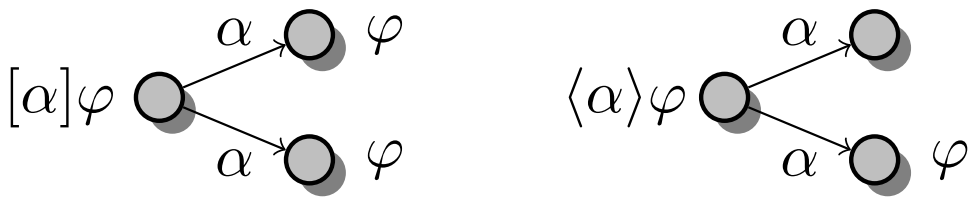
A branching-time language...



The Language

$$\begin{array}{rllllll} \varphi, \psi \in \mu\text{HML} ::= & \mathtt{tt} & \mid \langle\alpha\rangle\varphi & \mid \varphi \vee \psi & \mid \min X.\varphi & \mid X \\ & \mid \mathtt{ff} & \mid [\alpha]\varphi & \mid \varphi \wedge \psi & \mid \max X.\varphi & \end{array}$$

A branching-time language...



...or, possibly, a linear-time language...

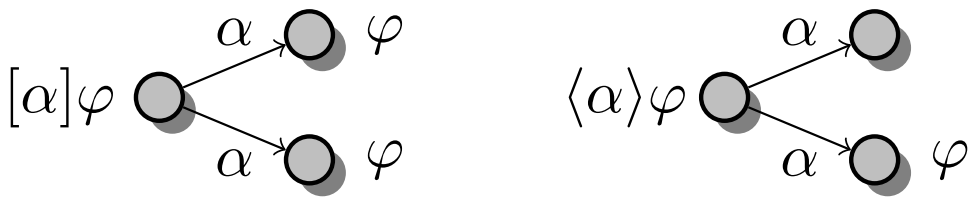
$$s \models [\alpha]\varphi : \quad s = \alpha\ s' \implies s' \models \varphi$$

$$s \models \langle\alpha\rangle\varphi : \quad s = \alpha\ s' \ \ \mathit{and} \ \ s' \models \varphi$$

The Language

$$\varphi, \psi \in \mu\text{HML} ::= \texttt{tt} \mid \langle\alpha\rangle\varphi \mid \varphi \vee \psi \mid \min X.\varphi \mid X$$
$$\mid \texttt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \max X.\varphi$$

A branching-time language...



...or, possibly, a linear-time language...

$$s \models [\alpha]\varphi : \quad s = \alpha\ s' \implies s' \models \varphi$$
$$s \models \langle\alpha\rangle\varphi : \quad s = \alpha\ s' \ \textit{and}\ \ s' \models \varphi$$

...with recursion

$\min X.\varphi$ holds iff $\varphi[\min X.\varphi/X]$ does and is a lfp

$\max X.\varphi$ holds iff $\varphi[\max X.\varphi/X]$ does and is a gfp

The Language

$$\varphi, \psi \in \mu\text{HML} ::= \texttt{tt} \mid \langle\alpha\rangle\varphi \mid \varphi \vee \psi \mid \min X.\varphi \mid X$$
$$\mid \texttt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \max X.\varphi$$

An *expressive* language:

can encode LTL, CTL, CTL*, BA,...

Shorthands from LTL on infinite traces:

$$\mathsf{X}\ \varphi := [\textsc{Act}]\varphi \qquad \text{(next step)}$$
$$\mathsf{F}\ \varphi := \min Y.(\varphi \vee \mathsf{X}\ Y) \qquad \text{(in the future)}$$
$$\mathsf{G}\ \varphi := \max Y.(\varphi \wedge \mathsf{X}\ Y) \qquad \text{(generaly)}$$
$$\varphi\ \mathsf{U}\ \psi := \min Y.(\psi \vee (\varphi \wedge \mathsf{X}\ Y)) \qquad \text{(until)}$$

Regular Monitors

Syntax:

$$m, n \in \textsc{Mon} ::= \texttt{end} \mid \texttt{no} \mid \alpha.m \mid m + n \mid \texttt{rec}\ x.m \mid x$$

Regular Monitors

Syntax:

$$m, n \in \textsc{Mon} ::= \texttt{end} \mid \texttt{no} \mid \alpha.m \mid m + n \mid \texttt{rec}\ x.m \mid x$$

Monitor LTS (verdicts are irrevocable):

$$\textsc{Act}\ \frac{}{\alpha.m \xrightarrow{\alpha} m} \qquad \textsc{Rec}\ \frac{m[\texttt{rec}x.m/x] \xrightarrow{\alpha} n}{\texttt{rec}x.m \xrightarrow{\alpha} n} \qquad \textsc{Vrd}\ \frac{}{\texttt{no} \xrightarrow{\alpha} \texttt{no}}$$

$$\textsc{SelL}\ \frac{m \xrightarrow{\alpha} m'}{m + n \xrightarrow{\alpha} m'} \qquad \textsc{SelR}\ \frac{n \xrightarrow{\alpha} n'}{m + n \xrightarrow{\alpha} n'} \qquad \textsc{Vrd}\ \frac{}{\texttt{end} \xrightarrow{\alpha} \texttt{end}}$$

Regular Monitors

Syntax:

$$m, n \in \textsc{Mon} ::= \texttt{end} \mid \texttt{no} \mid \alpha.m \mid m + n \mid \texttt{rec}\ x.m \mid x$$

Monitor LTS (verdicts are irrevocable):

$$\textsc{Act}\frac{}{\alpha.m \xrightarrow{\alpha} m} \qquad \textsc{Rec}\frac{m[\texttt{rec}x.m/x] \xrightarrow{\alpha} n}{\texttt{rec}x.m \xrightarrow{\alpha} n} \qquad \textsc{Vrd}\frac{}{\texttt{no} \xrightarrow{\alpha} \texttt{no}}$$

$$\textsc{SelL}\frac{m \xrightarrow{\alpha} m'}{m + n \xrightarrow{\alpha} m'} \qquad \textsc{SelR}\frac{n \xrightarrow{\alpha} n'}{m + n \xrightarrow{\alpha} n'} \qquad \textsc{Vrd}\frac{}{\texttt{end} \xrightarrow{\alpha} \texttt{end}}$$

Instrumentation (follow the trace):

$$\textsc{iMon}\frac{p \xrightarrow{\alpha}_L q \quad m \xrightarrow{\alpha}_M n}{m \triangleleft p \xrightarrow{\alpha}_I n \triangleleft q} \qquad \textsc{iTer}\frac{p \xrightarrow{\alpha}_L q \quad m \overset{\alpha}{\not\longrightarrow}_M}{m \triangleleft p \xrightarrow{\alpha}_I \texttt{end} \triangleleft q}$$

Formulas and Rejection-Monitors

- Formulas specify process properties: $p \models \varphi$
- Monitors run along a process and read its trace:
  $$m \triangleleft p \xrightarrow{\alpha_1} n \triangleleft q \xrightarrow{\alpha_2} \cdots$$
- A monitor can reach three possible verdicts: `yes`, `no`, `end`
- m accepts p when $m \triangleleft p \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_r} \texttt{yes} \triangleleft q$
  m rejects p when $m \triangleleft p \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_r} \texttt{no} \triangleleft q$
  for some $\alpha_1 \cdots \alpha_r$ and q
- `end` is the inconclusive verdict.

Definition (Complete Monitorability)

m monitors completely for φ when

- m accepts exactly the processes that satisfy φ; and
- m rejects exactly the ones that do not satisfy φ.

Complete Monitorability is Impossible

say m accepts p and rejects q

Complete Monitorability is Impossible

say m accepts p and rejects q

$p + q$ can produce all the traces of p and q

Complete Monitorability is Impossible

say m accepts p and rejects q

$p + q$ can produce all the traces of p and q

m must both accept and reject $p + q$

Complete Monitorability is Impossible

say m accepts p and rejects q

$p + q$ can produce all the traces of p and q

m must both accept and reject $p + q$

a sound monitor can either accept or reject, but not both

Definition (Partial Monitorability)

- m is *sound* (s.) for φ if it only accepts (rejects) processes that satisfy (violate) φ;
- m is *satisfaction-complete* (s.c.) for φ when m accepts all the processes that satisfy φ; and
- m is *violation-complete* (v.c.) for φ when m rejects all the processes that do not satisfy φ.

Complete Monitorability is Impossible

say m accepts p and rejects q

$p + q$ can produce all the traces of p and q

m must both accept and reject $p + q$

a sound monitor can either accept or reject, but not both

Definition (Partial Monitorability)

- m is *sound* (s.) for φ if it only accepts (rejects) processes that satisfy (violate) φ;
- m is *satisfaction-complete* (s.c.) for φ when m accepts all the processes that satisfy φ; and
- m is *violation-complete* (v.c.) for φ when m rejects all the processes that do not satisfy φ. We focus on violation.

Monitors for μHML (on processes)

FAI17, AAFI17

We can monitor for sHML, the safety fragment of μHML:

$$\varphi, \psi \in \text{sHML} ::= \texttt{tt} \mid \texttt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \max X.\varphi \mid X.$$

Monitor Synthesis Function: $\varphi \mapsto m(\varphi)$ (follow the syntax).
Formula Synthesis Function: $m \mapsto f(m)$.

Theorem (Monitorability – Maximality)

$m(\varphi)$ monitors for φ and m monitors for $f(m)$.
The basic monitoring system monitors for sHML.

Monitors for μHML (on processes)

FAI17, AAFI17

We can monitor for sHML, the safety fragment of μHML:

$$\varphi, \psi \in \text{sHML} ::= \texttt{tt} \mid \texttt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \max X.\varphi \mid X.$$

Monitor Synthesis Function: $\varphi \mapsto m(\varphi)$ (follow the syntax).
Formula Synthesis Function: $m \mapsto f(m)$.

Theorem (Monitorability – Maximality)

$m(\varphi)$ monitors for φ and m monitors for $f(m)$.
The basic monitoring system monitors for sHML.

Question: Are violation-monitorable properties closed under disjunction?

Monitors for μHML (on processes)

FAI17, AAFI17

We can monitor for sHML, the safety fragment of μHML:

$$\varphi, \psi \in \text{sHML} ::= \texttt{tt} \mid \texttt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \max X.\varphi \mid X.$$

Monitor Synthesis Function: $\varphi \mapsto m(\varphi)$ (follow the syntax).
Formula Synthesis Function: $m \mapsto f(m)$.

Theorem (Monitorability – Maximality)

$m(\varphi)$ monitors for φ and m monitors for $f(m)$.
The basic monitoring system monitors for sHML.

Question: Are violation-monitorable properties closed under disjunction? $[\alpha]\texttt{ff} \vee [\beta]\texttt{ff}$

Monitors for μHML (on processes)

FAI17, AAFI17

We can monitor for sHML, the safety fragment of μHML:

$$\varphi, \psi \in \text{sHML} ::= \ \mathtt{tt} \ \mid \mathtt{ff} \ \mid [\alpha]\varphi \ \mid \varphi \wedge \psi \ \mid \max X.\varphi \ \mid X.$$

Monitor Synthesis Function: $\varphi \mapsto m(\varphi)$ (follow the syntax).
Formula Synthesis Function: $m \mapsto f(m)$.

Theorem (Monitorability – Maximality)

$m(\varphi)$ monitors for φ and m monitors for $f(m)$.
The basic monitoring system monitors for sHML.

Question: Are violation-monitorable properties closed under disjunction? $[\alpha]\mathtt{ff} \vee [\beta]\mathtt{ff}$

Question: How about diamonds? $\langle\alpha\rangle\mathtt{tt}$

On Infinite Traces

$$\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \cdots$$

Are v-monitorable properties closed under $\vee$? $[\alpha]\mathtt{ff} \vee [\beta]\mathtt{ff}$

On Infinite Traces

$$\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \cdots$$

Are v-monitorable properties closed under $\vee$?

$$[\alpha]\mathtt{ff} \vee [\beta]\mathtt{ff}$$

$$\mathsf{G}\,[\alpha]\mathtt{ff} \vee \mathsf{G}\,[\beta]\mathtt{ff}$$

On Infinite Traces

$$\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \cdots$$

Are v-monitorable properties closed under $\vee$? $[\alpha]\mathtt{ff} \vee [\beta]\mathtt{ff}$

$\mathsf{G}\,[\alpha]\mathtt{ff} \vee \mathsf{G}\,[\beta]\mathtt{ff}$

How about diamonds? $\langle\alpha\rangle\mathtt{tt}$

On Infinite Traces

$$\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \cdots$$

Are v-monitorable properties closed under $\vee$? $[\alpha]\mathtt{ff} \vee [\beta]\mathtt{ff}$

$\mathsf{G}\,[\alpha]\mathtt{ff} \vee \mathsf{G}\,[\beta]\mathtt{ff}$

How about diamonds? $\langle\alpha\rangle\mathtt{tt}$

How about lfp? $\mathsf{F}\,\langle\alpha\rangle\mathtt{tt}$

Parallel monitors: we introduce two parallel operators, $\otimes$ and $\oplus$

$$\frac{m \xrightarrow{\alpha} m' \quad n \xrightarrow{\alpha} n'}{m \odot n \xrightarrow{\alpha} m' \odot n'} \qquad \frac{m \xrightarrow{\tau} m'}{m \odot n \xrightarrow{\tau} m' \odot n} \qquad \frac{}{\mathtt{end} \odot \mathtt{end} \xrightarrow{\tau} \mathtt{end}}$$

$$\frac{}{\mathtt{yes} \otimes m \xrightarrow{\tau} m} \qquad \frac{}{\mathtt{no} \otimes m \xrightarrow{\tau} \mathtt{no}} \qquad \frac{}{\mathtt{no} \oplus m \xrightarrow{\tau} m} \qquad \frac{}{\mathtt{yes} \oplus m \xrightarrow{\tau} \mathtt{yes}}$$

On Infinite Traces

$$\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \cdots$$

Are v-monitorable properties closed under $\vee$? $[\alpha]\mathtt{ff} \vee [\beta]\mathtt{ff}$

$\mathsf{G}\,[\alpha]\mathtt{ff} \vee \mathsf{G}\,[\beta]\mathtt{ff}$

How about diamonds? $\langle\alpha\rangle\mathtt{tt}$

How about lfp? $\mathsf{F}\,\langle\alpha\rangle\mathtt{tt}$

Parallel monitors: we introduce two parallel operators, $\otimes$ and $\oplus$

$$\frac{m \xrightarrow{\alpha} m' \quad n \xrightarrow{\alpha} n'}{m \odot n \xrightarrow{\alpha} m' \odot n'} \qquad \frac{m \xrightarrow{\tau} m'}{m \odot n \xrightarrow{\tau} m' \odot n} \qquad \frac{}{\mathtt{end} \odot \mathtt{end} \xrightarrow{\tau} \mathtt{end}}$$

$$\frac{}{\mathtt{yes} \otimes m \xrightarrow{\tau} m} \qquad \frac{}{\mathtt{no} \otimes m \xrightarrow{\tau} \mathtt{no}} \qquad \frac{}{\mathtt{no} \oplus m \xrightarrow{\tau} m} \qquad \frac{}{\mathtt{yes} \oplus m \xrightarrow{\tau} \mathtt{yes}}$$

Examples: $\mathtt{rec}\ x.(\alpha.\mathtt{no} + \overline{\alpha}.x) \oplus \mathtt{rec}\ x.(\beta.\mathtt{no} + \overline{\beta}.x)$,

On Infinite Traces

$$\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \cdots$$

Are v-monitorable properties closed under $\vee$? $[\alpha]\mathtt{ff} \vee [\beta]\mathtt{ff}$

$\mathsf{G}\,[\alpha]\mathtt{ff} \vee \mathsf{G}\,[\beta]\mathtt{ff}$

How about diamonds? $\langle\alpha\rangle\mathtt{tt}$

How about lfp? $\mathsf{F}\,\langle\alpha\rangle\mathtt{tt}$

Parallel monitors: we introduce two parallel operators, $\otimes$ and $\oplus$

$$\frac{m \xrightarrow{\alpha} m' \quad n \xrightarrow{\alpha} n'}{m \odot n \xrightarrow{\alpha} m' \odot n'} \qquad \frac{m \xrightarrow{\tau} m'}{m \odot n \xrightarrow{\tau} m' \odot n} \qquad \frac{}{\mathtt{end} \odot \mathtt{end} \xrightarrow{\tau} \mathtt{end}}$$

$$\frac{}{\mathtt{yes} \otimes m \xrightarrow{\tau} m} \qquad \frac{}{\mathtt{no} \otimes m \xrightarrow{\tau} \mathtt{no}} \qquad \frac{}{\mathtt{no} \oplus m \xrightarrow{\tau} m} \qquad \frac{}{\mathtt{yes} \oplus m \xrightarrow{\tau} \mathtt{yes}}$$

Examples: $\mathtt{rec}\ x.(\alpha.\mathtt{no} + \overline{\alpha}.x) \oplus \mathtt{rec}\ x.(\beta.\mathtt{no} + \overline{\beta}.x)$, $\quad \overline{\alpha}.\mathtt{no}$,

On Infinite Traces

$$\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \cdots$$

Are v-monitorable properties closed under $\vee$? $[\alpha]\mathtt{ff} \vee [\beta]\mathtt{ff}$

$\mathsf{G}\,[\alpha]\mathtt{ff} \vee \mathsf{G}\,[\beta]\mathtt{ff}$

How about diamonds? $\langle\alpha\rangle\mathtt{tt}$

How about lfp? $\mathsf{F}\,\langle\alpha\rangle\mathtt{tt}$

Parallel monitors: we introduce two parallel operators, $\otimes$ and $\oplus$

$$\frac{m \xrightarrow{\alpha} m' \quad n \xrightarrow{\alpha} n'}{m \odot n \xrightarrow{\alpha} m' \odot n'} \qquad \frac{m \xrightarrow{\tau} m'}{m \odot n \xrightarrow{\tau} m' \odot n} \qquad \frac{}{\mathtt{end} \odot \mathtt{end} \xrightarrow{\tau} \mathtt{end}}$$

$$\frac{}{\mathtt{yes} \otimes m \xrightarrow{\tau} m} \qquad \frac{}{\mathtt{no} \otimes m \xrightarrow{\tau} \mathtt{no}} \qquad \frac{}{\mathtt{no} \oplus m \xrightarrow{\tau} m} \qquad \frac{}{\mathtt{yes} \oplus m \xrightarrow{\tau} \mathtt{yes}}$$

Examples: $\mathtt{rec}\ x.(\alpha.\mathtt{no} + \overline{\alpha}.x) \oplus \mathtt{rec}\ x.(\beta.\mathtt{no} + \overline{\beta}.x)$, $\quad \overline{\alpha}.\mathtt{no}$,
$\mathtt{rec}\ x.(\mathtt{rec}\ y.(\alpha.\mathtt{no} + \overline{\alpha}.y) \oplus (\beta.\mathtt{no} + \gamma.x))$

Parallel Monitors and maxHML

The monitorable syntactic fragments of μHML are now larger:

$$\begin{aligned}
\varphi, \psi \in \text{maxHML} ::= \ & \mathtt{tt} \mid \langle\alpha\rangle\varphi \mid \varphi \vee \psi \mid X \\
& \mid \mathtt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \max X.\varphi \\
\varphi, \psi \in \text{minHML} ::= \ & \mathtt{tt} \mid \langle\alpha\rangle\varphi \mid \varphi \vee \psi \mid X \\
& \mid \mathtt{ff} \mid [\alpha]\varphi \mid \varphi \wedge \psi \mid \min X.\varphi
\end{aligned}$$

$$\begin{aligned}
m(\langle\alpha\rangle\varphi) &= \alpha.m(\varphi) + \overline{\alpha}.\mathtt{no} & m(\varphi \wedge \psi) &= m(\varphi) \otimes m(\psi) \\
m([\alpha]\varphi) &= \alpha.m(\varphi) + \overline{\alpha}.\mathtt{yes} & m(\varphi \vee \psi) &= m(\varphi) \oplus m(\psi)
\end{aligned}$$

Parallel Monitors and maxHML

The monitorable syntactic fragments of μHML are now larger:

$$\begin{array}{rrrrr}
\varphi, \psi \in \text{maxHML} ::= & \mathtt{tt} & \mid \langle\alpha\rangle\varphi & \mid \varphi \vee \psi & \mid X \\
& \mid \mathtt{ff} & \mid [\alpha]\varphi & \mid \varphi \wedge \psi & \mid \max X.\varphi \\
\varphi, \psi \in \text{minHML} ::= & \mathtt{tt} & \mid \langle\alpha\rangle\varphi & \mid \varphi \vee \psi & \mid X \\
& \mid \mathtt{ff} & \mid [\alpha]\varphi & \mid \varphi \wedge \psi & \mid \min X.\varphi
\end{array}$$

$$\begin{array}{ll}
m(\langle\alpha\rangle\varphi) = \alpha.m(\varphi) + \overline{\alpha}.\mathtt{no} & m(\varphi \wedge \psi) = m(\varphi) \otimes m(\psi) \\
m([\alpha]\varphi) = \alpha.m(\varphi) + \overline{\alpha}.\mathtt{yes} & m(\varphi \vee \psi) = m(\varphi) \oplus m(\psi)
\end{array}$$

Concerns:

Are parallel monitors a reasonable model?
infinite-state; more powerful than regular monitors(?)

Parallel Monitors and maxHML

The monitorable syntactic fragments of μHML are now larger:

$$\begin{array}{rrrrr} \varphi, \psi \in \text{maxHML} ::= \mathtt{tt} & \mid \langle\alpha\rangle\varphi & \mid \varphi \vee \psi & \mid X \\ \mid \mathtt{ff} & \mid [\alpha]\varphi & \mid \varphi \wedge \psi & \mid \max X.\varphi \\ \varphi, \psi \in \text{minHML} ::= \mathtt{tt} & \mid \langle\alpha\rangle\varphi & \mid \varphi \vee \psi & \mid X \\ \mid \mathtt{ff} & \mid [\alpha]\varphi & \mid \varphi \wedge \psi & \mid \min X.\varphi \end{array}$$

$$\begin{array}{ll} m(\langle\alpha\rangle\varphi) = \alpha.m(\varphi) + \overline{\alpha}.\mathtt{no} & m(\varphi \wedge \psi) = m(\varphi) \otimes m(\psi) \\ m([\alpha]\varphi) = \alpha.m(\varphi) + \overline{\alpha}.\mathtt{yes} & m(\varphi \vee \psi) = m(\varphi) \oplus m(\psi) \end{array}$$

Concerns:

Are parallel monitors a reasonable model?
infinite-state; more powerful than regular monitors(?)

maxHML $\cap$ minHML appears to be nontrivial

Regularization

Parallel monitors are convenient, but not more powerful:

Theorem (Regularization)

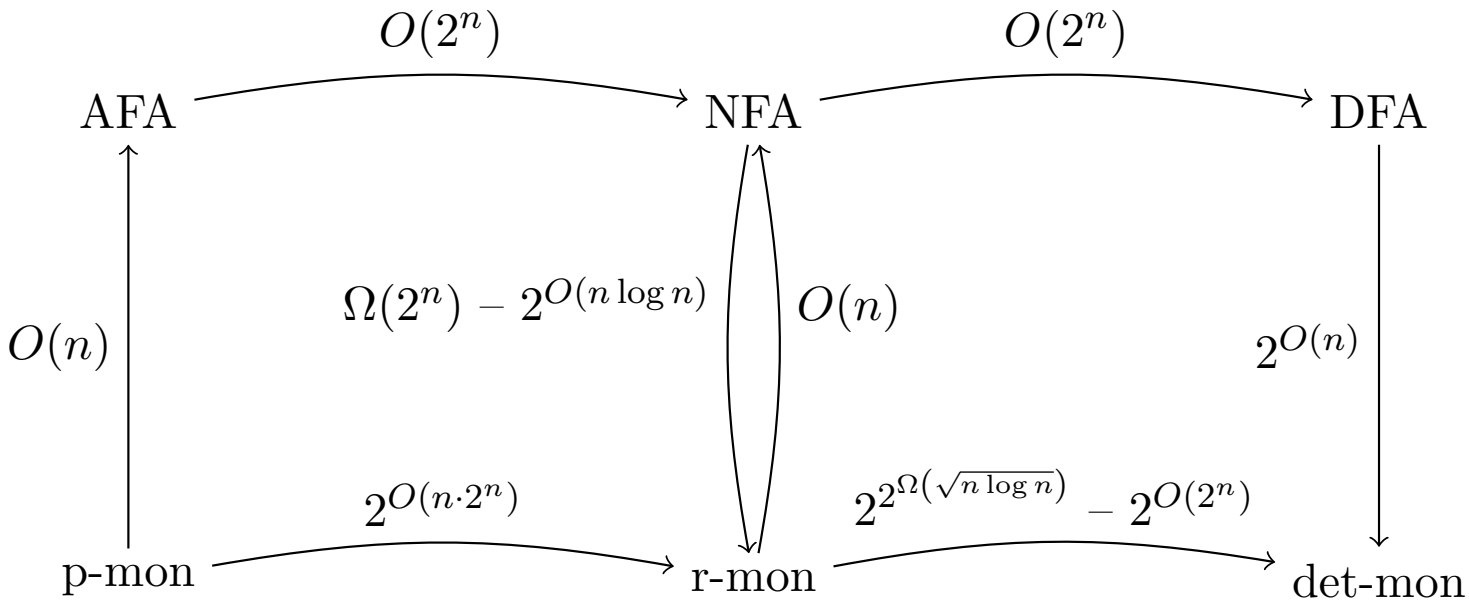
$m(\varphi)$ is equivalent to a regular monitor of size $2^{O(|m(\varphi)|\cdot 2^{|m(\varphi)|})}$.

Regularization

Parallel monitors are convenient, but not more powerful:

Theorem (Regularization)

$m(\varphi)$ is equivalent to a regular monitor of size $2^{O(|m(\varphi)|\cdot 2^{|m(\varphi)|})}$.



Surprise!

Complete monitorability *is* possible, after all

You cannot combine an accepting and a rejecting trace into one

$$m([\alpha][\beta]\mathtt{ff} \wedge [\beta][\alpha]\mathtt{ff}) \equiv \alpha.\beta.\mathtt{no} + \beta.\alpha.\mathtt{no} + \alpha.\alpha.\mathtt{yes} + \beta.\beta.\mathtt{yes}$$

Surprise!

Complete monitorability *is* possible, after all

You cannot combine an accepting and a rejecting trace into one

$$m([\alpha][\beta]\mathtt{ff} \wedge [\beta][\alpha]\mathtt{ff}) \equiv \alpha.\beta.\mathtt{no} + \beta.\alpha.\mathtt{no} + \alpha.\alpha.\mathtt{yes} + \beta.\beta.\mathtt{yes}$$

Theorem (Complete Monitorability)

For $\varphi \in HML$, $m(\varphi)$ monitors completely for φ over linear time.

Surprise!

Complete monitorability *is* possible, after all

You cannot combine an accepting and a rejecting trace into one

$$m([\alpha][\beta]\mathtt{ff} \wedge [\beta][\alpha]\mathtt{ff}) \equiv \alpha.\beta.\mathtt{no} + \beta.\alpha.\mathtt{no} + \alpha.\alpha.\mathtt{yes} + \beta.\beta.\mathtt{yes}$$

Theorem (Complete Monitorability)

For $\varphi \in HML$, $m(\varphi)$ monitors completely for φ over linear time.

Furthermore, all completely monitorable μHML trace-properties can be (constructively) written in HML — so, HML is (semantically) maximal.

Even Better: General Maximality of HML

All trace properties, irrespective of the monitoring system can be expressed in HML:

Theorem (General Maximality for HML)

Let m be a monitor from a monitoring system such that:

1. *verdicts are irrevocable: if m accepts/rejects a finite trace, then it accepts/rejects all its extensions, and*
2. *m accepts/rejects t iff, it accepts/rejects some finite prefix.*

For any property φ with a trace interpretation (not necessarily written in μHML), if m is sound and complete for φ then φ can be expressed in HML.

Proof sketch.

Every trace either satisfies φ or not, so m accepts or rejects. By König's Lemma, there is a finite set of finite traces that determine φ. Describe these in HML. □

Maximality for Partial Monitorability

and a collapse

Theorem (Partial Monitorability for Linear-Time)

For $\varphi \in$ maxHML (minHML), $m(\varphi)$ is sound and violation-(satisfaction-)complete for φ.

Maximality for Partial Monitorability

and a collapse

Theorem (Partial Monitorability for Linear-Time)

For $\varphi \in$ maxHML (minHML), $m(\varphi)$ is sound and violation-(satisfaction-)complete for φ. These fragments are maximal.

Proof of maximality.

We can transform:

m to regular n

n to $f(n) \in \text{sHML} \subseteq \text{maxHML}$. □

Maximality for Partial Monitorability

and a collapse

Theorem (Partial Monitorability for Linear-Time)

For $\varphi \in$ maxHML (minHML), $m(\varphi)$ is sound and violation-(satisfaction-)complete for φ. These fragments are maximal.

Proof of maximality.

We can transform:

m to regular n

n to $f(n) \in \text{sHML} \subseteq \text{maxHML}$. □

Corollary

$\text{maxHML} \equiv \text{sHML}$ *and* $\text{minHML} \equiv \text{cHML}$ *over traces.*

Maximality for Partial Monitorability

and a collapse

Theorem (Partial Monitorability for Linear-Time)

For $\varphi \in$ maxHML (minHML), $m(\varphi)$ is sound and violation-(satisfaction-)complete for φ. These fragments are maximal.

Proof of maximality.

We can transform:

m to regular n

n to $f(n) \in \text{sHML} \subseteq \text{maxHML}$. □

Corollary

$\text{maxHML} \equiv \text{sHML}$ *and* $\text{minHML} \equiv \text{cHML}$ *over traces.*

Question: General maximality for maxHML?

Maximality for Partial Monitorability

and a collapse

Theorem (Partial Monitorability for Linear-Time)

For $\varphi \in$ maxHML (minHML), $m(\varphi)$ is sound and violation-(satisfaction-)complete for φ. These fragments are maximal.

Proof of maximality.

We can transform:

m to regular n

n to $f(n) \in \text{sHML} \subseteq \text{maxHML}$. □

Corollary

$\text{maxHML} \equiv \text{sHML}$ *and* $\text{minHML} \equiv \text{cHML}$ *over traces.*

Question: General maximality for maxHML?

No: μHML properties are (ω-)regular and we can monitor with (say) PDAs.

How fast does your monitor return a verdict?

$\langle a \rangle [a][b]\mathtt{tt}$ $\qquad$ $b.\mathtt{no} + a.(b.\mathtt{yes} + a.(a.\mathtt{yes} + b.\mathtt{yes}))$ $\qquad$ $a\ a\ b\ a\ \cdots$

How fast does your monitor return a verdict?

$\langle a \rangle [a][b]\mathtt{tt}$ $\qquad$ $b.\mathtt{no} + a.(b.\mathtt{yes} + a.(a.\mathtt{yes} + b.\mathtt{yes}))$ $\qquad$ $a\ a\ b\ a\ \cdots$

Definition

A monitor m is *tight* when for every finite s, if m rejects all infinite extensions of s, then m rejects s.

We want to construct tight monitors.

Constructing Tight Monitors for HML

Slim formulas:

$$\varphi ::= \mathtt{tt} \mid \mathtt{ff} \mid \bigwedge_{\alpha \in B} [\alpha]\varphi_\alpha \mid \bigvee_{\alpha \in D} \langle\alpha\rangle\psi\alpha,$$

where $B, D \neq \emptyset$, $\varphi_\alpha \neq \mathtt{tt}$, $\psi_\alpha \neq \mathtt{ff}$, and no $\bigwedge_{\alpha \in \text{Act}} [\alpha]\mathtt{ff}$ or $\bigvee_{\alpha \in \text{Act}} \langle\alpha\rangle\mathtt{tt}$

(i.e. nothing is immediately true or false, except $\mathtt{tt}$ and $\mathtt{ff}$).

Constructing Tight Monitors for HML

Slim formulas:

$$\varphi ::= \mathtt{tt} \mid \mathtt{ff} \mid \bigwedge_{\alpha \in B} [\alpha]\varphi_\alpha \mid \bigvee_{\alpha \in D} \langle\alpha\rangle\psi\alpha,$$

where $B, D \neq \emptyset$, $\varphi_\alpha \neq \mathtt{tt}$, $\psi_\alpha \neq \mathtt{ff}$, and no $\bigwedge_{\alpha \in \mathrm{ACT}} [\alpha]\mathtt{ff}$ or $\bigvee_{\alpha \in \mathrm{ACT}} \langle\alpha\rangle\mathtt{tt}$

(i.e. nothing is immediately true or false, except $\mathtt{tt}$ and $\mathtt{ff}$).

Proposition

If $\varphi \in HML$ is slim, then $m(\varphi)$ is tight.

Constructing Tight Monitors for HML

Slim formulas:

$$\varphi ::= \mathtt{tt} \mid \mathtt{ff} \mid \bigwedge_{\alpha \in B} [\alpha]\varphi_\alpha \mid \bigvee_{\alpha \in D} \langle\alpha\rangle\psi\alpha,$$

where $B, D \neq \emptyset$, $\varphi_\alpha \neq \mathtt{tt}$, $\psi_\alpha \neq \mathtt{ff}$, and no $\bigwedge_{\alpha \in \mathrm{ACT}}[\alpha]\mathtt{ff}$ or $\bigvee_{\alpha \in \mathrm{ACT}}\langle\alpha\rangle\mathtt{tt}$

(i.e. nothing is immediately true or false, except $\mathtt{tt}$ and $\mathtt{ff}$).

Proposition

If $\varphi \in HML$ is slim, then $m(\varphi)$ is tight.

The dieting process is based on rewrite rules based on simple equivalences: $[\mathrm{ACT}]\mathtt{ff} \Rrightarrow \mathtt{ff}$, $\langle\mathrm{ACT}\rangle\mathtt{tt} \Rrightarrow \mathtt{tt}$, $\langle\alpha\rangle\mathtt{ff} \Rrightarrow \mathtt{ff}$, $[\alpha]\varphi \wedge [\alpha]\psi \Rrightarrow [\alpha](\varphi \wedge \psi)$, $[\alpha]\varphi \vee [\beta]\psi \Rrightarrow \mathtt{tt}$, $\langle\alpha\rangle\varphi \wedge [\beta]\psi \Rrightarrow \langle\alpha\rangle\varphi$ ...

Constructing Tight Monitors for maxHML

Transform φ to $m(\varphi)$, and then

determinize $m(\varphi)$ to m;

Constructing Tight Monitors for maxHML

Transform φ to $m(\varphi)$, and then

determinize $m(\varphi)$ to m;

replace: $\texttt{rec}\ x.\texttt{no} \Rrightarrow \texttt{no}$, $\sum_{\alpha \in \mathrm{ACT}} \alpha.\texttt{no} \Rrightarrow \texttt{no}$,
$\texttt{rec}\ x.\texttt{yes} \Rrightarrow \texttt{yes}$, and $\sum_{\alpha \in \mathrm{ACT}} \alpha.\texttt{yes} \Rrightarrow \texttt{yes}$,

until there is nothing to replace.

Proposition

The result of the above process is tight.

Constructing Tight Monitors for maxHML

Transform φ to $m(\varphi)$, and then

determinize $m(\varphi)$ to m;

replace: $\texttt{rec}\ x.\texttt{no} \Rrightarrow \texttt{no}$, $\sum_{\alpha \in \mathrm{ACT}} \alpha.\texttt{no} \Rrightarrow \texttt{no}$,
$\texttt{rec}\ x.\texttt{yes} \Rrightarrow \texttt{yes}$, and $\sum_{\alpha \in \mathrm{ACT}} \alpha.\texttt{yes} \Rrightarrow \texttt{yes}$,

until there is nothing to replace.

Proposition

The result of the above process is tight.

The resulting monitor can be triple-exponentially larger than φ.

Constructing Tight Monitors for maxHML

Question: Can we have a nicer construction as for HML?

Constructing Tight Monitors for maxHML

Question: Can we have a nicer construction as for HML?

Probably not: `no` is the tight monitor for φ iff φ is not satisfiable.

Proposition (Upper bound from Vardi, 1988)

maxHML-*satisfiability on infinite traces is* PSPACE-*complete (for* $|\text{Act}| > 1$*).*

Into the Finfinite: $\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4 \cdots ?$

another possible model: finite or infinite traces

semantics similar to infinite traces

Into the Finfinite: $\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4 \cdots ?$

another possible model: finite or infinite traces

semantics similar to infinite traces

we lose complete monitorability

Into the Finfinite: $\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4 \cdots ?$

another possible model: finite or infinite traces

semantics similar to infinite traces

we lose complete monitorability

tightness becomes irrelevant

Into the Finfinite: $\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4 \cdots$?

another possible model: finite or infinite traces

semantics similar to infinite traces

we lose complete monitorability

tightness becomes irrelevant

$\langle\alpha\rangle\mathtt{tt}$ no longer monitorable for violation

Into the Finfinite: $\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4 \cdots$?

another possible model: finite or infinite traces

semantics similar to infinite traces

we lose complete monitorability

tightness becomes irrelevant

$\langle\alpha\rangle\mathtt{tt}$ no longer monitorable for violation

maximally monitorable fragments:

$$\begin{array}{rcccccccc}
\varphi,\psi \in \textsc{unHML} ::= & \mathtt{tt} & | & \mathtt{ff} & | & [\alpha]\varphi & & \\
| & \varphi \vee \psi & | & \varphi \wedge \psi & | & \max X.\varphi & | & X, \\
\varphi,\psi \in \textsc{exHML} ::= & \mathtt{tt} & | & \mathtt{ff} & | & \langle\alpha\rangle\varphi & & \\
| & \varphi \vee \psi & | & \varphi \wedge \psi & | & \min X.\varphi & | & X.
\end{array}$$

Into the Finfinite: $\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4 \cdots$?

another possible model: finite or infinite traces

semantics similar to infinite traces

we lose complete monitorability

tightness becomes irrelevant

$\langle\alpha\rangle\mathtt{tt}$ no longer monitorable for violation

maximally monitorable fragments:

$$\begin{array}{rcl}
\varphi,\psi \in \textsc{unHML} ::= & \mathtt{tt} \mid \mathtt{ff} \mid [\alpha]\varphi & \\
\mid & \varphi \vee \psi \mid \varphi \wedge \psi \mid \max X.\varphi \mid X, & \\
\varphi,\psi \in \textsc{exHML} ::= & \mathtt{tt} \mid \mathtt{ff} \mid \langle\alpha\rangle\varphi & \\
\mid & \varphi \vee \psi \mid \varphi \wedge \psi \mid \min X.\varphi \mid X. &
\end{array}$$

$\textsc{unHML} \equiv \textsc{sHML}$, $\textsc{exHML} \equiv \textsc{cHML}$ over finfinite traces

What we did

- Automatic monitor synthesis from l.t. maxHML, minHML
  - Either directly from maxHML or minHML, to construct a parallel monitor and then regularize, or from sHML or cHML, to directly give a regular monitor.
  - Can produce tight monitors.
  - For sHML, cHML, there are working tools (DetectEr). These can be used out-of-the box even for l.t.

What we did

- Automatic monitor synthesis from l.t. maxHML, minHML
  - Either directly from maxHML or minHML, to construct a parallel monitor and then regularize, or from sHML or cHML, to directly give a regular monitor.
  - Can produce tight monitors.
  - For sHML, cHML, there are working tools (DetectEr). These can be used out-of-the box even for l.t.
- Complete characterization of monitorable trace properties with respect to different monitorability guarantees.

What we did

- Automatic monitor synthesis from l.t. maxHML, minHML
  - Either directly from maxHML or minHML, to construct a parallel monitor and then regularize, or from sHML or cHML, to directly give a regular monitor.
  - Can produce tight monitors.
  - For sHML, cHML, there are working tools (DetectEr). These can be used out-of-the box even for l.t.
- Complete characterization of monitorable trace properties with respect to different monitorability guarantees.
- Logical consequences: maxHML collapses to sHML for l.t.

What we did

- Automatic monitor synthesis from l.t. maxHML, minHML
  - Either directly from maxHML or minHML, to construct a parallel monitor and then regularize, or from sHML or cHML, to directly give a regular monitor.
  - Can produce tight monitors.
  - For sHML, cHML, there are working tools (DetectEr). These can be used out-of-the box even for l.t.
- Complete characterization of monitorable trace properties with respect to different monitorability guarantees.
- Logical consequences: maxHML collapses to sHML for l.t.
- On the other hand, we see surprising differences
  - Complete monitorability, tightness
  - Monitorable formulas are closed under $\wedge, \vee$ for l.t., but not for branching time.

What we did

- Automatic monitor synthesis from l.t. maxHML, minHML
  - Either directly from maxHML or minHML, to construct a parallel monitor and then regularize, or from sHML or cHML, to directly give a regular monitor.
  - Can produce tight monitors.
  - For sHML, cHML, there are working tools (DetectEr). These can be used out-of-the box even for l.t.
- Complete characterization of monitorable trace properties with respect to different monitorability guarantees.
- Logical consequences: maxHML collapses to sHML for l.t.
- On the other hand, we see surprising differences
  - Complete monitorability, tightness
  - Monitorable formulas are closed under $\wedge, \vee$ for l.t., but not for branching time.
- Moral: when you study the properties of monitorability, the choice of the model matters!

Future work

- More complexity bounds
- What *is* monitorability, after all? Relation to other concepts, the rest of the RV community?
- Relations to similar concepts: diagnosability, learning,...
- Distributed RV, fault tolerance, Epistemic Logic, evidence tracking.
- Real time
- ...

More Variations for the Interested

- Silence and obscuring:
  - Sometimes we abstract away from internal system behaviour using a *silent* action τ.
  - Often, a silent transition is almost-visible due to evaluating conditions, system noise, or by design...
  - A framework for grades of *obscuring* of silent actions and *reliable* monitorability [AAFI17]
- Monitoring with conditions [AAFI18]
- Sound/optimal monitorability [Leh]

The End

Time for questions

Thank you for your attention

This research was supported by the projects "TheoFoMon: Theoretical Foundations for Monitorability" (grant number: 163406-051) and "Epistemic Logic for Distributed Runtime Monitoring" (grant number: 184940-051) of the Icelandic Research Fund, by the BMBF project "AramisII" (project number: 01IS160253), and by the EPSRC project "Solving parity games in theory and practice" (project number: EP/P020909/1).

Short Bibliography for Further Reading

The TheoFoMon project page: `http://icetcs.ru.is/theofomon/`

This work will appear in POPL 2019.

Luca Aceto, Antonis Achilleos, Adrian Francalanza, and Anna Ingólfsdóttir, *Monitoring for silent actions*, FSTTCS (Dagstuhl, Germany), LIPIcs, vol. 93, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, pp. 7:1–7:14.

Luca Aceto, Antonis Achilleos, Adrian Francalanza, and Anna Ingólfsdóttir, *A framework for parametrized monitorability*, FOSSACS, Lecture Notes in Computer Science, vol. 10803, Springer, 2018, pp. 203–220.

Karoliina Lehtinen, *Runtime verification of fixpoint logic: Synthesis of optimal monitors*, `https://www.informatik.uni-kiel.de/~leh/mon.pdf`.