From First-order Temporal Logic to Parametric Trace Slicing

Giles Reger David Rydeheard

University of Manchester, Manchester, UK

September 25, 2015

Outline

Motivation

- There are lots and lots of languages used for specifying RV properties (see the competition)
- Particularly for first-order/parametric/data properties
  - Whilst propositional case seems well understood, lots more freedom with first-order
  - Mainly how to organise the domain of quantification
  - Languages often driven by monitoring concerns
- We should understand how they are related

- Parametric trace slicing can be efficiently monitored
- Temporal logic is well understand and widely used
- If we can understand their connection we can leverage both advantages

Interpreting Formulas

- Does this trace
- satisfy this formula
- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace
- satisfy this formula

$$\forall x : \Box(p(x) \rightarrow \bigcirc q(x))$$

- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace

$$p(a).p(b).q(a).q(b).p(c).q(c).p(d).q(d)$$

- satisfy this formula

$$\forall x : \Box(p(x) \rightarrow \bigcirc q(x))$$

- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace

$$p(a).p(b).q(a).q(b).p(c).q(c).p(d).q(d)$$

- satisfy this formula

$$\forall x : \Box(p(x) \rightarrow \bigcirc q(x))$$

- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace

$$p(a).p(b).q(a).q(b).p(c).q(c).p(d).q(d)$$

- satisfy this formula

$$\forall x : \Box(p(x) \rightarrow \bigcirc q(x))$$

- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace

$$p(a).p(b).q(a).q(b).p(c).q(c).p(d).q(d)$$

- satisfy this formula

$$\forall x : \neg q(x)\, \mathcal{U}\; p(x)$$

- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace

$$\texttt{open}(A).\texttt{open}(B).\texttt{open}(B).\texttt{close}(A).\texttt{close}(A)$$

- satisfy this formula

$$\forall f : \texttt{open}(f) \rightarrow (\neg\texttt{open}(f)\,\mathcal{U}^{\circ}\,\texttt{close}(f))$$

- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace

$$\texttt{open}(A).\texttt{open}(B).\texttt{open}(B).\texttt{close}(A).\texttt{close}(A)$$

- satisfy this formula

$$\forall f : \texttt{open}(f) \rightarrow (\neg\texttt{open}(f)\, \mathcal{U}^{\circ}\, \texttt{close}(f))$$

- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace

$$\texttt{open}(A).\texttt{open}(B).\texttt{open}(B).\texttt{close}(A).\texttt{close}(A)$$

- satisfy this formula

$$\forall f : \texttt{open}(f) \rightarrow (\neg\texttt{open}(f)\,\mathcal{U}^{\circ}\,\texttt{close}(f))$$

- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?

Interpreting Formulas

- Does this trace
- satisfy this formula
- In the *'standard'* view of quantification?
- In the *'slicing'* view of quantification?
- Other notions of quantification exist that give different interpretations, we stick to these two for now

Introducing FO-LTL$_f$

- Time is linear, discrete and future
- Finite-trace semantics
- Syntax (note use of next-Until)

$$\phi = \mathit{true} \mid \mathbf{a} \mid \forall x : \phi \mid \neg\phi \mid \phi \vee \phi \mid \phi\, \mathcal{U}^{\circ}\, \phi$$

- Semantics

$$\begin{array}{lcll}
\mathcal{D}, \tau, v, i & \models & \mathit{true} & \\
\mathcal{D}, \tau, v, i & \models & \mathbf{a} & \text{if } \tau_i = v(\mathbf{a}) \\
\mathcal{D}, \tau, v, i & \models & \neg\phi & \text{if } \mathcal{D}, \tau, v, i \not\models \phi \\
\mathcal{D}, \tau, v, i & \models & \phi_1 \vee \phi_2 & \text{if } \mathcal{D}, \tau, v, i \models \phi_1 \text{ or } \mathcal{D}, \tau, v, i \models \phi_2 \\
\mathcal{D}, \tau, v, i & \models & \phi_1\, \mathcal{U}^{\circ}\, \phi_2 & \text{if there exists a } j > i \text{ such that either} \\
 & & & \quad \mathcal{D}, \tau, v, j \models \phi_2 \text{ or } (j = |\tau| \text{ and } \phi_2 = \mathit{false}) \\
 & & & \text{and for} i < k < j \text{ we have } \mathcal{D}, \tau, v, k \models \phi_1 \\
\mathcal{D}, \tau, v, i & \models & \forall x : \phi & \text{if for every } d \in \mathcal{D}(x) \text{ we have} \\
 & & & \mathcal{D}, \tau, v \dagger [x \mapsto d], i \models \phi
\end{array}$$

Introducing FO-LTL$_f$

- Time is linear, discrete and future
- Finite-trace semantics
- Syntax (note use of next-Until)

$$\phi = \mathit{true} \mid \mathbf{a} \mid \forall x : \phi \mid \neg\phi \mid \phi \vee \phi \mid \phi\, \mathcal{U}^{\circ}\, \phi$$

- Semantics

$$\begin{array}{lcll}
\mathcal{D}, \tau, v, i & \models & \mathit{true} & \\
\mathcal{D}, \tau, v, i & \models & \mathbf{a} & \text{if } \tau_i = v(\mathbf{a}) \\
\mathcal{D}, \tau, v, i & \models & \neg\phi & \text{if } \mathcal{D}, \tau, v, i \not\models \phi \\
\mathcal{D}, \tau, v, i & \models & \phi_1 \vee \phi_2 & \text{if } \mathcal{D}, \tau, v, i \models \phi_1 \text{ or } \mathcal{D}, \tau, v, i \models \phi_2 \\
\mathcal{D}, \tau, v, i & \models & \phi_1\, \mathcal{U}^{\circ}\, \phi_2 & \text{if there exists a } j > i \text{ such that either} \\
 & & & \quad \mathcal{D}, \tau, v, j \models \phi_2 \text{ or } (j = |\tau| \text{ and } \phi_2 = \mathit{false}) \\
 & & & \text{and for} i < k < j \text{ we have } \mathcal{D}, \tau, v, k \models \phi_1 \\
\mathcal{D}, \tau, v, i & \models & \forall x : \phi & \text{if for every } d \in \mathcal{D}(x) \text{ we have} \\
 & & & \mathcal{D}, \tau, v \dagger [x \mapsto d], i \models \phi
\end{array}$$

Introducing FO-LTL$_f$

- Time is linear, discrete and future
- Finite-trace semantics
- Syntax (note use of next-Until)

$$\phi = \mathit{true} \mid \mathbf{a} \mid \forall x : \phi \mid \neg\phi \mid \phi \vee \phi \mid \phi\, \mathcal{U}^{\circ}\, \phi$$

- Semantics

$$
\begin{array}{lcll}
\mathcal{D}, \tau, v, i & \models & \mathit{true} & \\
\mathcal{D}, \tau, v, i & \models & \mathbf{a} & \text{if } \tau_i = v(\mathbf{a}) \\
\mathcal{D}, \tau, v, i & \models & \neg\phi & \text{if } \mathcal{D}, \tau, v, i \not\models \phi \\
\mathcal{D}, \tau, v, i & \models & \phi_1 \vee \phi_2 & \text{if } \mathcal{D}, \tau, v, i \models \phi_1 \text{ or } \mathcal{D}, \tau, v, i \models \phi_2 \\
\mathcal{D}, \tau, v, i & \models & \phi_1\, \mathcal{U}^{\circ}\, \phi_2 & \text{if there exists a } j > i \text{ such that either} \\
 & & & \quad \mathcal{D}, \tau, v, j \models \phi_2 \text{ or } (j = |\tau| \text{ and } \phi_2 = \mathit{false}) \\
 & & & \text{and for} i < k < j \text{ we have } \mathcal{D}, \tau, v, k \models \phi_1 \\
\mathcal{D}, \tau, v, i & \models & \forall x : \phi & \text{if for every } d \in \mathcal{D}(x) \text{ we have} \\
 & & & \mathcal{D}, \tau, v \dagger [x \mapsto d], i \models \phi
\end{array}
$$

Definitions

- Can define the normal things in terms of $\mathcal{U}^\circ$

$$
\begin{array}{lcl}
\bigcirc\varphi & = & \mathit{false}\,\mathcal{U}^\circ\ \varphi \\
\phi_1\,\mathcal{U}\,\phi_2 & = & \phi_2 \vee (\phi_1 \wedge (\phi_1\,\mathcal{U}^\circ\ \phi_2)) \\
\Diamond\phi & = & \mathit{true}\,\mathcal{U}\ \phi \\
\Box\phi & = & \phi\,\mathcal{U}\ \mathit{false}
\end{array}
$$

- Next is strong i.e. $\bigcirc$**a** is false at the end of the trace
- But $\Box$**a** will be true at the end of the trace
- And $\Diamond$**a** will be false at the end of the trace

- Slightly non-standard finite trace semantics, would like to vary in the future

Domain of quantification

- The (other) controversial bit

- We write $\tau \models \phi$ if a trace τ satisfies a property ϕ, defined as follows

$$\tau \models \phi \quad \text{iff} \quad \mathtt{dom}(\tau,\phi), \tau, [], 0 \models \phi$$

where the domain function $\mathtt{dom}$ is defined as:

$$\mathtt{dom}(\tau,\phi)(x) = \left\{ d_i \text{ where } \begin{array}{l} e(\ldots, d_i, \ldots) \in \tau \wedge \\ e(\ldots, x_i, \ldots) \in \mathtt{events}(\phi) \wedge \\ x_i = x \end{array} \right\}$$

- The domain of quantification is dependent on the full trace

Parametric Trace Slicing

- Given a trace τ and valuation θ let $\tau\downarrow_\theta$ be the θ-slice of τ

$$\begin{array}{lcl} \epsilon\downarrow_\theta & = & \epsilon \\ \tau.e(\overline{v})\downarrow_\theta & = & \begin{cases} (\tau\downarrow_\theta).e(\overline{v}) & \text{if } \exists e(\overline{z}) \in \mathcal{A}(X) : \theta(e(\overline{z})) = e(\overline{v}) \\ (\tau\downarrow_\theta) & \text{otherwise} \end{cases} \end{array}$$

- The trace τ is accepted for quantification list $\Lambda(X)$ and propositional property $\mathcal{P}(X)$ if $\tau \models_{[]}^{\mathcal{P}(X)} \Lambda(X)$, defined as

$$\begin{array}{ll} \tau \models_\theta^{\mathcal{P}(X)} \forall x : \Lambda & \text{if for every } d \in \mathtt{dom}(x) \text{ we have } \tau \models_{\theta\dagger[x\mapsto d]}^{\mathcal{P}(X)} \Lambda \\ \tau \models_\theta^{\mathcal{P}(X)} \exists x : \Lambda & \text{if for some } d \in \mathtt{dom}(x) \text{ we have } \tau \models_{\theta\dagger[x\mapsto d]}^{\mathcal{P}(X)} \Lambda \\ \tau \models_\theta^{\mathcal{P}(X)} \epsilon & \text{if } \tau\downarrow_\theta \in \mathcal{L}(\theta, \mathcal{P}(X)) \end{array}$$

- Using the same domain of quantification `dom`

Example

Given the trace

$\texttt{call}(A).\texttt{call}(B).\texttt{call}(C).\texttt{return}(C).\texttt{return}(B).\texttt{call}(C).\texttt{return}(C).\texttt{return}(A)$

And a property φ that whenever a method m_2 is called inside a method m_1, the method m_2 should return before m_1.

$$\texttt{events}(\varphi) = \{\texttt{call}(m_1), \texttt{return}(m_1), \texttt{call}(m_2), \texttt{return}(m_2)\}$$

We get the following slices

m_1	m_2	slice
A	B	$\texttt{call}(A).\texttt{call}(B).\texttt{return}(B).\texttt{return}(A)$
A	C	$\texttt{call}(A).\texttt{call}(C).\texttt{return}(C).\texttt{call}(C).\texttt{return}(C).\texttt{return}(A)$
B	C	$\texttt{call}(B).\texttt{call}(C).\texttt{return}(C).\texttt{return}(B).\texttt{call}(C).\texttt{return}(C)$

Each slice can be checked by some unquantified checker $\texttt{P}(\texttt{m}_1, \texttt{m}_2)$

Goal of considering Sliceability

- Identify the properties of FO-LTL$_f$ formulas that allow the semantics to coincide with the slicing semantics

- For now only consider *globally quantified properties*
- This is an artificial restriction (as our slicing definition is restricted) that makes everything easier for now
- ... but the globally quantified fragment is still interesting
- .. and we have begun to consider the full setting

- Aim to understand correspondence
- And use this to efficiently monitor FO-LTL$_f$ properties

Slicing Invariance

A formula ψ with free variables X is *sliceable* if for valuation θ over X and trace τ. The formula ψ is *sliceable* if

$$\tau, \theta \models \psi \quad \Leftrightarrow \quad \tau \downarrow_\theta, \theta \models \psi.$$

Let $\mathcal{L}(\psi, \theta) = \{\tau \mid \tau, \theta \models \psi\}$ be the traces satisfying ψ. Define $\mathcal{L}^C(\psi, \theta)$ as the *non-relevance-closure* of $\mathcal{L}(\psi, \theta)$ to be the smallest set containing

$$\begin{array}{lll}
\tau & \text{if} & \tau \in \mathcal{L}(\psi, \theta) \\
\tau_1.\tau_2.\tau_3 & \text{if} & \forall \mathbf{a} \in \tau_2 : \mathbf{a} \notin \text{relevant}(\psi, \theta) \text{ and } \tau_1.\tau_3 \in \mathcal{L}^C(\psi, \theta) \\
\tau_1.\tau_3 & \text{if} & \exists \tau_1.\tau_2.\tau_3 \in \mathcal{L}^C(\psi, \theta) : \forall \mathbf{a} \in \tau_2 : \mathbf{a} \notin \text{relevant}(\psi, \theta)
\end{array}$$

where $\text{relevant}(\psi, \theta) = \{\theta(\mathbf{a}) \mid \mathbf{a} \in \text{events}(\psi)\}$. The formula ψ is *slicing invariant* if $\mathcal{L}(\psi, \theta) = \mathcal{L}^C(\psi, \theta)$.

The notions of sliceability and slicing invariance coincide.

What are the restrictions?

$$\mathtt{k}(a).\mathtt{f}(a).\mathtt{f}(b).\mathtt{g}(a).\mathtt{g}(b).\mathtt{h}(b,a).\mathtt{f}(b).\mathtt{h}(b,c).\mathtt{g}(b)$$

Starting at the start

- $\mathtt{f}(x) \vee \Diamond\mathtt{k}(x)$ and $\neg\mathtt{f}(x) \vee \Diamond\mathtt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg\mathtt{f}(x)\,\mathcal{U}^{\circ}\,\mathtt{g}(x)$ for the trace $\mathtt{g}(a).\mathtt{g}(a).\mathtt{f}(b).\mathtt{g}(b)$.

Never saying next

- $\Box(\mathtt{f}(x) \rightarrow \bigcirc\mathtt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\mathtt{f}(x) \vee \mathtt{g}(x)) = (\mathtt{f}(x) \vee \mathtt{g}(x))\,\mathcal{U}\,\mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg\mathtt{f}(x) \wedge \neg\mathtt{g}(x)) = \mathit{true}\,\mathcal{U}\,(\neg\mathtt{f}(x) \wedge \neg\mathtt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg\varphi_2$

What are the restrictions?

$$\mathtt{k}(a).\mathtt{f}(a).\mathtt{f}(b).\mathtt{g}(a).\mathtt{g}(b).\mathtt{h}(b,a).\mathtt{f}(b).\mathtt{h}(b,c).\mathtt{g}(b)$$
$$.\quad .\mathtt{f}(b).\quad .\mathtt{g}(b).\quad .\mathtt{f}(b).\quad .\mathtt{g}(b)$$

Starting at the start

- $\mathtt{f}(x) \vee \Diamond \mathtt{k}(x)$ and $\neg \mathtt{f}(x) \vee \Diamond \mathtt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg \mathtt{f}(x)\, \mathcal{U}^\circ\, \mathtt{g}(x)$ for the trace $\mathtt{g}(a).\mathtt{g}(a).\mathtt{f}(b).\mathtt{g}(b)$.

Never saying next

- $\Box(\mathtt{f}(x) \rightarrow \bigcirc \mathtt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\mathtt{f}(x) \vee \mathtt{g}(x)) = (\mathtt{f}(x) \vee \mathtt{g}(x))\, \mathcal{U}\, \mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg \mathtt{f}(x) \wedge \neg \mathtt{g}(x)) = \mathit{true}\, \mathcal{U}\, (\neg \mathtt{f}(x) \wedge \neg \mathtt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg \varphi_2$

What are the restrictions?

$$\texttt{k}(a).\texttt{f}(a).\texttt{f}(b).\texttt{g}(a).\texttt{g}(b).\texttt{h}(b,a).\texttt{f}(b).\texttt{h}(b,c).\texttt{g}(b)$$
$$.\quad .\texttt{f}(b).\quad .\texttt{g}(b).\quad .\texttt{f}(b).\quad .\texttt{g}(b)$$

Starting at the start

- $\texttt{f}(x) \vee \Diamond\texttt{k}(x)$ and $\neg\texttt{f}(x) \vee \Diamond\texttt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg\texttt{f}(x)\,\mathcal{U}^\circ\ \texttt{g}(x)$ for the trace $\texttt{g}(a).\texttt{g}(a).\texttt{f}(b).\texttt{g}(b)$.

Never saying next

- $\Box(\texttt{f}(x) \rightarrow \bigcirc\texttt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\texttt{f}(x) \vee \texttt{g}(x)) = (\texttt{f}(x) \vee \texttt{g}(x))\,\mathcal{U}\ \mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg\texttt{f}(x) \wedge \neg\texttt{g}(x)) = \mathit{true}\,\mathcal{U}\ (\neg\texttt{f}(x) \wedge \neg\texttt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg\varphi_2$

What are the restrictions?

$$\texttt{k}(a).\texttt{f}(a).\texttt{f}(b).\texttt{g}(a).\texttt{g}(b).\texttt{h}(b,a).\texttt{f}(b).\texttt{h}(b,c).\texttt{g}(b)$$
$$. \quad .\texttt{f}(b). \quad .\texttt{g}(b). \quad .\texttt{f}(b). \quad .\texttt{g}(b)$$

Starting at the start

- $\texttt{f}(x) \vee \Diamond\texttt{k}(x)$ and $\neg\texttt{f}(x) \vee \Diamond\texttt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg\texttt{f}(x)\, \mathcal{U}^\circ\, \texttt{g}(x)$ for the trace $\texttt{g}(a).\texttt{g}(a).\texttt{f}(b).\texttt{g}(b)$.

Never saying next

- $\Box(\texttt{f}(x) \rightarrow \bigcirc\texttt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\texttt{f}(x) \vee \texttt{g}(x)) = (\texttt{f}(x) \vee \texttt{g}(x))\, \mathcal{U}\, \mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg\texttt{f}(x) \wedge \neg\texttt{g}(x)) = \mathit{true}\, \mathcal{U}\, (\neg\texttt{f}(x) \wedge \neg\texttt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg\varphi_2$

What are the restrictions?

$$\mathtt{k}(a).\mathtt{f}(a).\mathtt{f}(b).\mathtt{g}(a).\mathtt{g}(b).\mathtt{h}(b,a).\mathtt{f}(b).\mathtt{h}(b,c).\mathtt{g}(b)$$
$$. \quad .\mathtt{f}(b). \quad .\mathtt{g}(b). \quad .\mathtt{f}(b). \quad .\mathtt{g}(b)$$

Starting at the start

- $\mathtt{f}(x) \vee \Diamond\mathtt{k}(x)$ and $\neg\mathtt{f}(x) \vee \Diamond\mathtt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg\mathtt{f}(x)\,\mathcal{U}^\circ\,\mathtt{g}(x)$ for the trace $\mathtt{g}(a).\mathtt{g}(a).\mathtt{f}(b).\mathtt{g}(b)$.

Never saying next

- $\Box(\mathtt{f}(x) \rightarrow \bigcirc\mathtt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\mathtt{f}(x) \vee \mathtt{g}(x)) = (\mathtt{f}(x) \vee \mathtt{g}(x))\,\mathcal{U}\,\mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg\mathtt{f}(x) \wedge \neg\mathtt{g}(x)) = \mathit{true}\,\mathcal{U}\,(\neg\mathtt{f}(x) \wedge \neg\mathtt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg\varphi_2$

What are the restrictions?

$$\mathtt{k}(a).\mathtt{f}(a).\mathtt{f}(b).\mathtt{g}(a).\mathtt{g}(b).\mathtt{h}(b,a).\mathtt{f}(b).\mathtt{h}(b,c).\mathtt{g}(b)$$
$$.\quad .\mathtt{f}(b).\quad .\mathtt{g}(b).\quad .\mathtt{f}(b).\quad .\mathtt{g}(b)$$

Starting at the start

- $\mathtt{f}(x) \vee \Diamond\mathtt{k}(x)$ and $\neg\mathtt{f}(x) \vee \Diamond\mathtt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg\mathtt{f}(x)\,\mathcal{U}^{\circ}\,\mathtt{g}(x)$ for the trace $\mathtt{g}(a).\mathtt{g}(a).\mathtt{f}(b).\mathtt{g}(b)$.

Never saying next

- $\Box(\mathtt{f}(x) \rightarrow \bigcirc\mathtt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\mathtt{f}(x) \vee \mathtt{g}(x)) = (\mathtt{f}(x) \vee \mathtt{g}(x))\,\mathcal{U}\,\mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg\mathtt{f}(x) \wedge \neg\mathtt{g}(x)) = \mathit{true}\,\mathcal{U}\,(\neg\mathtt{f}(x) \wedge \neg\mathtt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg\varphi_2$

What are the restrictions?

$$\mathtt{k}(a).\mathtt{f}(a).\mathtt{f}(b).\mathtt{g}(a).\mathtt{g}(b).\mathtt{h}(b,a).\mathtt{f}(b).\mathtt{h}(b,c).\mathtt{g}(b)$$
$$.\quad .\mathtt{f}(b).\quad .\mathtt{g}(b).\quad .\mathtt{f}(b).\quad .\mathtt{g}(b)$$

Starting at the start

- $\mathtt{f}(x) \vee \Diamond\mathtt{k}(x)$ and $\neg\mathtt{f}(x) \vee \Diamond\mathtt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg\mathtt{f}(x)\,\mathcal{U}^{\circ}\,\mathtt{g}(x)$ for the trace $\mathtt{g}(a).\mathtt{g}(a).\mathtt{f}(b).\mathtt{g}(b)$.

Never saying next

- $\Box(\mathtt{f}(x) \rightarrow \bigcirc\mathtt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\mathtt{f}(x) \vee \mathtt{g}(x)) = (\mathtt{f}(x) \vee \mathtt{g}(x))\,\mathcal{U}\,\mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg\mathtt{f}(x) \wedge \neg\mathtt{g}(x)) = \mathit{true}\,\mathcal{U}\,(\neg\mathtt{f}(x) \wedge \neg\mathtt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg\varphi_2$

What are the restrictions?

$$\mathtt{k}(a).\mathtt{f}(a).\mathtt{f}(b).\mathtt{g}(a).\mathtt{g}(b).\mathtt{h}(b,a).\mathtt{f}(b).\mathtt{h}(b,c).\mathtt{g}(b)$$
$$.\quad .\mathtt{f}(b). \quad .\mathtt{g}(b). \quad .\mathtt{f}(b). \quad .\mathtt{g}(b)$$

Starting at the start

- $\mathtt{f}(x) \vee \Diamond \mathtt{k}(x)$ and $\neg \mathtt{f}(x) \vee \Diamond \mathtt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg \mathtt{f}(x)\, \mathcal{U}^\circ\, \mathtt{g}(x)$ for the trace $\mathtt{g}(a).\mathtt{g}(a).\mathtt{f}(b).\mathtt{g}(b)$.

Never saying next

- $\Box(\mathtt{f}(x) \rightarrow \bigcirc \mathtt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\mathtt{f}(x) \vee \mathtt{g}(x)) = (\mathtt{f}(x) \vee \mathtt{g}(x))\, \mathcal{U}\, \mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg \mathtt{f}(x) \wedge \neg \mathtt{g}(x)) = \mathit{true}\, \mathcal{U}\, (\neg \mathtt{f}(x) \wedge \neg \mathtt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg \varphi_2$

What are the restrictions?

$$\texttt{k}(a).\texttt{f}(a).\texttt{f}(b).\texttt{g}(a).\texttt{g}(b).\texttt{h}(b,a).\texttt{f}(b).\texttt{h}(b,c).\texttt{g}(b)$$
$$. \quad .\texttt{f}(b). \quad .\texttt{g}(b). \quad .\texttt{f}(b). \quad .\texttt{g}(b)$$

Starting at the start

- $\texttt{f}(x) \vee \Diamond\texttt{k}(x)$ and $\neg\texttt{f}(x) \vee \Diamond\texttt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg\texttt{f}(x)\, \mathcal{U}^\circ\, \texttt{g}(x)$ for the trace $\texttt{g}(a).\texttt{g}(a).\texttt{f}(b).\texttt{g}(b)$.

Never saying next

- $\Box(\texttt{f}(x) \rightarrow \bigcirc\texttt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\texttt{f}(x) \vee \texttt{g}(x)) = (\texttt{f}(x) \vee \texttt{g}(x))\, \mathcal{U}\, \mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg\texttt{f}(x) \wedge \neg\texttt{g}(x)) = \mathit{true}\, \mathcal{U}\, (\neg\texttt{f}(x) \wedge \neg\texttt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg\varphi_2$

What are the restrictions?

$$\texttt{k}(a).\texttt{f}(a).\texttt{f}(b).\texttt{g}(a).\texttt{g}(b).\texttt{h}(b,a).\texttt{f}(b).\texttt{h}(b,c).\texttt{g}(b)$$
$$.\quad .\texttt{f}(b).\quad .\texttt{g}(b).\quad .\texttt{f}(b).\quad .\texttt{g}(b)$$

Starting at the start

- $\texttt{f}(x) \vee \Diamond\texttt{k}(x)$ and $\neg\texttt{f}(x) \vee \Diamond\texttt{k}(x)$ i.e. for $x = b$
- Rule: Cannot allow events (in positive or negative form) at the top level of a sliceable formula
- $\neg\texttt{f}(x)\,\mathcal{U}^{\circ}\,\texttt{g}(x)$ for the trace $\texttt{g}(a).\texttt{g}(a).\texttt{f}(b).\texttt{g}(b)$.

Never saying next

- $\Box(\texttt{f}(x) \rightarrow \bigcirc\texttt{g}(x))$ for $x = b$
- $\varphi_1 = \Box(\texttt{f}(x) \vee \texttt{g}(x)) = (\texttt{f}(x) \vee \texttt{g}(x))\,\mathcal{U}\,\mathit{false}$ for $x = b$
- Rule: cannot restrict what happens at the next time point

Never saying never

- $\varphi_2 = \Diamond(\neg\texttt{f}(x) \wedge \neg\texttt{g}(x)) = \mathit{true}\,\mathcal{U}\,(\neg\texttt{f}(x) \wedge \neg\texttt{g}(x))$ for $x = b$
- Rule: cannot wait for the negation of things

Symmetry. Note that $\varphi_1 = \neg\varphi_2$

Define Syntactic Fragment $\mathcal{F}$

Let $\mathcal{F}$ be those formulas

$$Q_1 x_1 : \ldots Q_n x_n : \psi_T$$

for zero or more quantifications $Q_i x_i$, with $Q_i = \forall$ or $\exists$, and quantifier-free ψ_T inductively defined as :

$$\begin{array}{lcl}
\psi_T & = & \psi_L \,\mathcal{U}\, \psi_R \mid \psi_T \vee \psi_T \mid \psi_T \wedge \psi_T \\
\psi_L & = & \mathit{true} \mid \psi_L \vee \psi_U \mid \psi_L \wedge \psi_L \mid \neg \mathbf{a} \\
\psi_R & = & \mathit{false} \mid \psi_R \wedge \psi_U \mid \psi_R \vee \psi_R \mid \mathbf{a} \\
\psi_U & = & \psi_L \,\mathcal{U}^{\circ}\, \psi_R \mid \psi_L \,\mathcal{U}\, \psi_R \mid \psi_U \vee \psi_U \mid \psi_U \wedge \psi_U
\end{array}$$

- Negations only on atoms
- Left formulas (ψ_L) are true on non-relevant events
- Right formulas (ψ_R) are false on non-relevant events

Properties

Lemma

For any formula $\psi \in \mathcal{F}$ with free variables X, traces τ_1 and τ_2, valuations θ over X, events $\mathbf{a} \notin \mathrm{relevant}(\psi, \theta)$ and indices i, j:

Case 1. *If ψ in $\psi_{L,R,U}$ and $(\tau_1.\mathbf{a}.\tau_2)_i \in \mathrm{relevant}(\psi, \theta)$ then*

$$\tau_1.\mathbf{a}.\tau_2, \theta, i \models \psi \Leftrightarrow \tau_1.\tau_2, \theta, j \models \psi$$

$$\text{where } j = \begin{cases} i & \text{if } i < |\tau_1| \\ j = i - 1 & \text{otherwise} \end{cases}$$

Case 2. *If ψ is in ψ_L then $\tau_1.\mathbf{a}.\tau_2, \theta, |\tau_1| \models \psi$*

Case 3. *If ψ is in ψ_R then $\tau_1.\mathbf{a}.\tau_2, \theta, |\tau_1| \not\models \psi$*

Case 4. *If ψ is in ψ_T then $\tau_1.\mathbf{a}.\tau_2, \theta, 0 \models \psi \Leftrightarrow \tau_1.\tau_2, \theta, 0 \models \psi$*

Theorem

All formulas in $\mathcal{F}$ are sliceable.

Is $\mathcal{F}$ usable as a specification language?

- Most properties in Dwyer et al. fit, obviously some do not
- HasNext

$$\begin{aligned}\forall i :\quad & (\neg\texttt{next}(i)\,\mathcal{U}\,\texttt{hasNext}(i))\wedge \\ & \Box(\texttt{next}(i) \rightarrow (\neg\texttt{next}(i)\,\mathcal{U}^{\circ}\,\texttt{hasNext}(i)))\end{aligned}$$

- UnsafeMapIter

$$\begin{aligned}\forall m : \forall c : \forall i :\quad & \Box(\texttt{create}(m, c) \rightarrow \Box(\texttt{iterator}(c, i) \rightarrow \\ & \quad \Box(\texttt{update}(m) \rightarrow \Box\neg\texttt{use}(i))))\end{aligned}$$

- CallNesting

$$\begin{aligned}&\forall m_1 : \forall m_2 : \\ &\ (\neg\texttt{ret}(m_1)\,\mathcal{U}\,\texttt{call}(m_1)) \wedge (\neg\texttt{ret}(m_2)\,\mathcal{U}\,\texttt{call}(m_2))\wedge \\ &\quad \Box(\texttt{call}(m_1) \rightarrow (\neg\texttt{call}(m_1)\,\mathcal{U}\,\texttt{ret}(m_1))) \wedge \Box(\texttt{call}(m_1) \rightarrow \\ &\quad (\texttt{call}(m_2) \rightarrow ((\neg\texttt{ret}(m_2) \wedge \neg\texttt{call}(m_2))\,\mathcal{U}\,\texttt{ret}(m_2)))\,\mathcal{U}\,\texttt{ret}(m_1))\end{aligned}$$

From $\mathcal{F}$ to slicing-based formalism QEA

- Now we have defined $\mathcal{F}$ we can translate formulas in $\mathcal{F}$ to a formalism that can be efficiently monitored
- We choose QEA as this is our formalism

- Straightforward progression-based translation of quantifier-free part to automaton
- Technique is not new but we couldn't find it written down nicely anywhere

Example

Consider HasNext

$$\psi = (\mathbf{h} \vee (\mathbf{n} \wedge (\neg \mathbf{n}\, \mathcal{U}^\circ\, \mathbf{h}))) \wedge (\neg \mathbf{n} \vee (\neg \mathbf{n}\, \mathcal{U}^\circ\, \mathbf{h})) \wedge ((\neg \mathbf{n} \vee (\neg \mathbf{n}\, \mathcal{U}^\circ\, \mathbf{h}))\, \mathcal{U}^\circ\, F)$$

	$\xrightarrow{\mathbf{h}}$	$\xrightarrow{\mathbf{n}}$
$\phi_1 = \mathbf{n}$	*false*	*true*
$\phi_2 = \mathbf{h}$	*true*	*false*
$\phi_3 = \neg\phi_1\, \mathcal{U}^\circ\, \phi_2$	ϕ_4	ϕ_4
$\phi_4 = \phi_2 \vee (\neg\phi_1 \wedge \phi_3)$	*true*	*false*
$\phi_5 = \neg\phi_1 \vee \phi_3$	*true*	ϕ_4
$\phi_6 = \phi_5\, \mathcal{U}^\circ\, \mathit{false}$	$\phi_5 \wedge \phi_6$	$\phi_5 \wedge \phi_6$
$\psi = \phi_4 \wedge \phi_6$	$\phi_5 \wedge \phi_6$	*false*

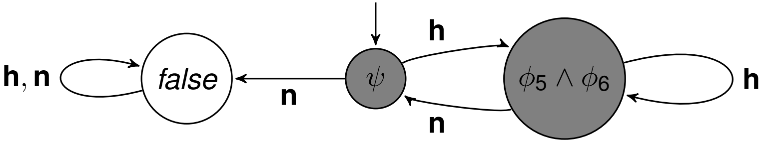
- $\phi_5 \wedge \phi_6 \xrightarrow{\mathbf{n}} (\phi_4 \wedge \phi_5 \wedge \phi_6) = \psi$
- We observe three states: ψ, *false* and $\phi_5 \wedge \phi_6$.
- Acceptance based on acceptance of empty trace
- $\phi_6 = \phi_5\, \mathcal{U}^\circ\, \mathit{false}$ is true on empty trace

Example

Consider HasNext

$$\psi = (\mathbf{h} \vee (\mathbf{n} \wedge (\neg \mathbf{n}\, \mathcal{U}^\circ\, \mathbf{h}))) \wedge (\neg \mathbf{n} \vee (\neg \mathbf{n}\, \mathcal{U}^\circ\, \mathbf{h})) \wedge ((\neg \mathbf{n} \vee (\neg \mathbf{n}\, \mathcal{U}^\circ\, \mathbf{h}))\, \mathcal{U}^\circ\, F)$$

- $\phi_5 \wedge \phi_6 \xrightarrow{\mathbf{n}} (\phi_4 \wedge \phi_5 \wedge \phi_6) = \psi$
- We observe three states: ψ, *false* and $\phi_5 \wedge \phi_6$.
- Acceptance based on acceptance of empty trace
- $\phi_6 = \phi_5\, \mathcal{U}^\circ$ *false* is true on empty trace



Further Work

- Is $\mathcal{F}$ a maximal fragment?
- Extensions
  - Other finite-trace semantics (multi-valued)
  - Arbitrary predicates
  - Non-global quantifiers
  - Freeze quantifiers
  - Translation in the other direction?
- Implement translation in MARQ

Conclusions

- First step in understanding the correspondence between two languages used for first-order runtime verification
- Important that we understand the specification language space
- Lots more to do