Towards a Case Study on Runtime Verification of Lightweight Avionic Controllers using RMTL-$\int_3$

André de Matos Pedro

CISTER ISEP/IPP & HASLab INESC TEC

November 25, 2015

Talk Outline

Motivation

Background

- Specification Language RMTL-$\int_3$
- Synthesis of *three-valued restricted metric temporal logic with durations* (RMTL-$\int_3$)
  - Simplification of Quantified Formulas
  - Temporal operators and duration terms as higher order functions
- Stable and non-stable Conditions for Monitoring

Ongoing Case Study based on PixHawk autopilot

- Platform
- RunTime Embedded Monitoring Library – RTEML
- Languages and Synthesizer

Talk Outline

Motivation I

Why is dynamic property verification and enforcement required for real-time systems?

- Real-time systems have a high dependence of temporal and timed constraints.
- Techniques for runtime verification have been growing progressively along the past years as a complement of static approaches such as deductive verification and model checking.
- **The demand of reliable and safe development of programs for embedded applications where time is crucial.**

Motivation I

Why is dynamic property verification and enforcement required for real-time systems?

- Real-time systems have a high dependence of temporal and timed constraints.
- Techniques for runtime verification have been growing progressively along the past years as a complement of static approaches such as deductive verification and model checking.
- **The demand of reliable and safe development of programs for embedded applications where time is crucial.**

*** AND THEY ARE MISSING ***

A lack of timed approaches for hard real-time properties of embedded system controllers. Not just a matter of functional properties.

Motivation II

What is dynamic property verification and enforcement of real-time systems?

- an approach for **schedulability analysis** of hard real-time systems and detection of **anomalies**;
- an approach to ensure **durations** of the past executions;
- a **complementary approach** of the common static analysis relaxing coverage but increasing the type of properties to be checked;

Motivation II

What is dynamic property verification and enforcement of real-time systems?

- an approach for **schedulability analysis** of hard real-time systems and detection of **anomalies**;
- an approach to ensure **durations** of the past executions;
- a **complementary approach** of the common static analysis relaxing coverage but increasing the type of properties to be checked;

*** AND IT IS ***

Motivation III

A lightweight method that can **reduce the burden** *for software* **programmers** *as well as intensive test cases. Maintainability and integration is also cover.* **Monitors run indeterminately** *awaiting the best* **stamp**.

Talk Outline

Warm up

- Language for specification of timed and resource constraints for real-time systems.

Warm up

- Language for specification of timed and resource constraints for real-time systems.
- Sound and reliable automatic monitor generation algorithm (synthesis).

Warm up

- Language for specification of timed and resource constraints for real-time systems.
- Sound and reliable automatic monitor generation algorithm (synthesis).
- Reliable Runtime Embedded Monitoring Library (RTEML) for ARM architecture.

Talk Outline

Duration aware specification

RMTL-$\int_3$ is formed by

- Terms:

$$\eta \ ::= \ \alpha \mid x \mid \min_{x \in I} \varphi \mid \ \max_{x \in I} \varphi \mid \eta_1 \circ \eta_2 \mid \int^{\eta} \varphi$$

- Formulas:

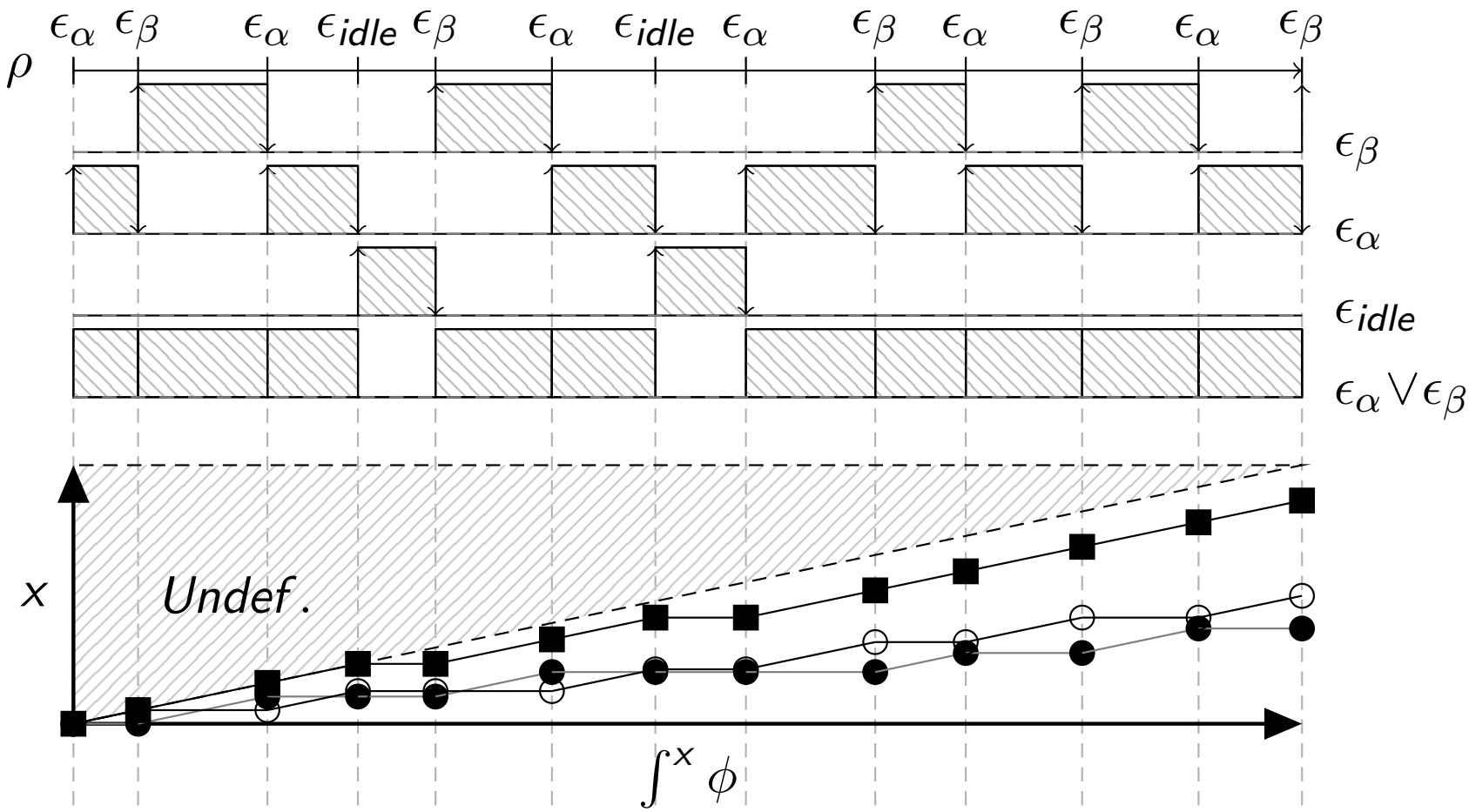
$$\varphi \ ::= \ p \mid \eta_1 < \eta_2 \mid \varphi_1 \vee \varphi_2 \mid \neg\varphi \mid \varphi_1 \, U_{\sim\gamma} \, \varphi_2 \mid \varphi_1 \, S_{\sim\gamma} \, \varphi_2 \mid \exists x \, \varphi,$$

with $\alpha \in \mathbb{R}$, $x \in \mathcal{V}$, $\circ$ a function $\{+, \times\}$, $\int^{\eta} \varphi$ the duration of the formula φ in an interval, $p \in \mathcal{P}$, $\sim \in \{<, =\}$, $\gamma \in \mathbb{R}_{\geq 0}$, and $<$ a relation.

Common Abbreviations

Eventually	:	$\Diamond_{\sim\gamma}\phi$	$\equiv$	$\mathit{true} \ U_{\sim\gamma} \ \phi$
Always	:	$\Box_{\sim\gamma}\phi$	$\equiv$	$\neg(\Diamond_{\sim\gamma}\neg\phi)$

Intuition for duration terms



- ρ is a path; ϵ_β, ϵ_α, and ϵ_{idle} are events;
- $\phi = \epsilon_\beta$ (●—), $\phi = \epsilon_\alpha$ (○—), and $\phi = \epsilon_\beta \vee \epsilon_\alpha$ (■—)

Talk Outline

Monitoring Synthesis of RMTL-$\int_3$

Intuition behind monitor generation

1. Quantifiers are discarded prior the execution. Any monitoring formula is quantifier free.

$$\exists x\ A \rightarrow (B \wedge (x < 10)),$$

which is trivially simplified to $A \rightarrow B$.

Monitoring Synthesis of RMTL-$\int_3$

Intuition behind monitor generation

1. Quantifiers are discarded prior the execution. Any monitoring formula is quantifier free.

$$\exists x\ A \rightarrow (B \wedge (x < 10)),$$

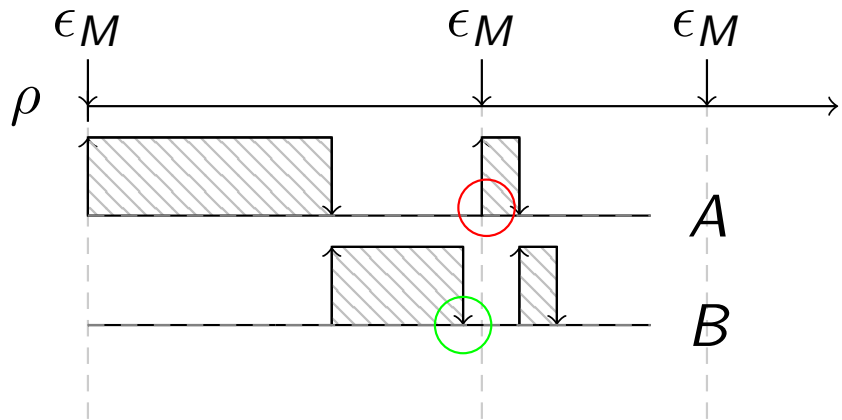
which is trivially simplified to $A \rightarrow B$.

2. Temporal operators such as until (U) and since (S) are encoded using higher order functions.

$A\ U_{<20}\ B$ is true for $t = 0$

$A\ U_{<21}\ B$ is false for $t = 0$

ϵ_M is released at each 20 time units, and the trace ρ begins at $t = 0$.



Simplification of RMTL-$\int_3$ formulas I

Definition (Inequality Abstraction Constraint)

Let ϕ_3 be a formula in RMTL-$\int_3$. ϕ_3 is a formula in *first order logic of real numbers* ($\text{FOL}_{\mathbb{R}}$) if it is free of duration terms, minimum/maximum terms, temporal operators, and propositions.

Simplification of RMTL-$\int_3$ formulas I

Definition (Inequality Abstraction Constraint)

Let ϕ_3 be a formula in RMTL-$\int_3$. ϕ_3 is a formula in *first order logic of real numbers* ($\text{FOL}_{\mathbb{R}}$) if it is free of duration terms, minimum/maximum terms, temporal operators, and propositions.

Step by step example of the proposed simplification algorithm

1. $x < \int^{x+1} (P \wedge x < 10)$
   {replace duration term by y}

Simplification of RMTL-$\int_3$ formulas I

Definition (Inequality Abstraction Constraint)

Let ϕ_3 be a formula in RMTL-$\int_3$. ϕ_3 is a formula in *first order logic of real numbers* ($\text{FOL}_{\mathbb{R}}$) if it is free of duration terms, minimum/maximum terms, temporal operators, and propositions.

Step by step example of the proposed simplification algorithm

1. $x < \int^{x+1} (P \wedge x < 10)$
   {replace duration term by y}
2. $x < y \wedge 0 \leq y \leq x + 1$
   {apply weaker inequality for $P \wedge x < 10$ }

Simplification of RMTL-$\int_3$ formulas I

Definition (Inequality Abstraction Constraint)

Let ϕ_3 be a formula in RMTL-$\int_3$. ϕ_3 is a formula in *first order logic of real numbers* ($\mathrm{FOL}_{\mathbb{R}}$) if it is free of duration terms, minimum/maximum terms, temporal operators, and propositions.

Step by step example of the proposed simplification algorithm

1. $x < \int^{x+1} (P \wedge x < 10)$
   {replace duration term by y}
2. $x < y \wedge 0 \leq y \leq x + 1$
   {apply weaker inequality for $P \wedge x < 10$ }
3. $x < y \wedge 0 \leq y \leq x + 1 \wedge$
   $\Big((x < 10) \rightarrow \Big(0 \leq \int^{x+1} P \leq x + 1\Big)\Big) \wedge$
   $\neg(x < 10) \rightarrow \mathrm{ff}$
   {replace new duration term by z}

Simplification of RMTL-$\int_3$ formulas II

4. $x < y \wedge 0 \leq y \leq x + 1 \wedge$
   $((x < 10) \rightarrow (0 \leq z \leq x + 1)) \wedge$
   $\neg(x < 10) \rightarrow \mathsf{ff}$
   {apply CAD }

Simplification of RMTL-$\int_3$ formulas II

4. $x < y \wedge 0 \leq y \leq x + 1 \wedge$
   $((x < 10) \rightarrow (0 \leq z \leq x + 1)) \wedge$
   $\neg(x < 10) \rightarrow \mathrm{ff}$
   {apply CAD }
5. $y = 0 \wedge (z = 0 \vee (0 \leq z \leq x + 1))) \vee$
   $(0 < y \leq x + 1 \wedge 0 \leq z \leq x + 1)$ for $x \in [-1, 0[$, and
   $(x < y \leq x + 1 \wedge 0 \leq z \leq x + 1)$ for $x \in [0, 10[$
   {replace y and z by $\int^{x+1} P$ }

Simplification of RMTL-$\int_3$ formulas II

4. $x < y \wedge 0 \leq y \leq x + 1 \wedge$
   $((x < 10) \rightarrow (0 \leq z \leq x + 1)) \wedge$
   $\neg(x < 10) \rightarrow \mathrm{ff}$
   {apply CAD }
5. $y = 0 \wedge (z = 0 \vee (0 \leq z \leq x + 1))) \vee$
   $(0 < y \leq x + 1 \wedge 0 \leq z \leq x + 1)$ for $x \in [-1, 0[$, and
   $(x < y \leq x + 1 \wedge 0 \leq z \leq x + 1)$ for $x \in [0, 10[$
   {replace y and z by $\int^{x+1} P$ }
6. $\int^{x+1} P = 0 \vee 0 < \int^{x+1} P \leq x + 1$ for $x \in [-1, 0[$,
   $x < \int^{x+1} P \leq x + 1$ for $x \in [0, 10[$, and
   ff otherwise

Simplification of RMTL-$\int_3$ formulas II

4. $x < y \wedge 0 \leq y \leq x + 1 \wedge$
   $((x < 10) \rightarrow (0 \leq z \leq x + 1)) \wedge$
   $\neg(x < 10) \rightarrow \mathrm{ff}$
   {apply CAD }
5. $y = 0 \wedge (z = 0 \vee (0 \leq z \leq x + 1))) \vee$
   $(0 < y \leq x + 1 \wedge 0 \leq z \leq x + 1)$ for $x \in [-1, 0[$, and
   $(x < y \leq x + 1 \wedge 0 \leq z \leq x + 1)$ for $x \in [0, 10[$
   {replace y and z by $\int^{x+1} P$ }
6. $\int^{x+1} P = 0 \vee 0 < \int^{x+1} P \leq x + 1$ for $x \in [-1, 0[$,
   $x < \int^{x+1} P \leq x + 1$ for $x \in [0, 10[$, and
   ff otherwise

Now, we have that $\forall x, x < \int^{x+1} (P \wedge x < 10)$ is false, and $\exists x, x < \int^{x+1} (P \wedge x < 10)$ is true.

Simplification of RMTL-$\int_3$ formulas III

After simplifying $\forall x, (0 \leq x < 10) \rightarrow x < \int^{x+1} (P \wedge x < 10)$, we have $\forall x, (0 \leq x < 10) \rightarrow 0 < \int^{x+1} P - x \leq x + 1$.

The proposition P is now isolated and we have two options,

Simplification of RMTL-$\int_3$ formulas III

After simplifying $\forall x, (0 \leq x < 10) \rightarrow x < \int^{x+1} (P \wedge x < 10)$, we have $\forall x, (0 \leq x < 10) \rightarrow 0 < \int^{x+1} P - x \leq x + 1$.

The proposition P is now isolated and we have two options,

1. replace x with a set of unique values – formula size highly increases (unpractical for real numbers search space)

Simplification of RMTL-$\int_3$ formulas III

After simplifying $\forall x, (0 \leq x < 10) \rightarrow x < \int^{x+1} (P \wedge x < 10)$, we have $\forall x, (0 \leq x < 10) \rightarrow 0 < \int^{x+1} P - x \leq x + 1$.

The proposition P is now isolated and we have two options,

1. replace x with a set of unique values – formula size highly increases (unpractical for real numbers search space)
2. increasingly compute duration terms for different intervals (uncovered in this paper) – may have a low monitoring overhead

Synthesis of duration term: $\int^{a} \phi$

$$\mathrm{Compute}_{(\int)}\ (\kappa, \upsilon)\ t\ a\ \phi \triangleq \begin{cases} ev^{\eta}_{al}\ (\kappa, \upsilon)\ \phi\ (sub_{\mathsf{K}}\ (\kappa, \upsilon, t)\ a) & \text{if } a \geq 0 \\ \bot_{\mathbb{R}} & \text{otherwise} \end{cases}$$

Synthesis of duration term: $\int^{a} \phi$

$$
\begin{aligned}
&ev^{\eta}_{al} && :: (\mathbf{K} \times \Upsilon) \to \Phi^{3} \to \mathbf{K} \to \mathbb{R}_{\geq 0} \\
&ev^{\eta}_{al}\ (\kappa, \upsilon)\ \phi\ \varkappa && \triangleq fold\ \left(\lambda s, (p, (i, t')) \to t' \cdot \left(1_{\varphi(\kappa,\upsilon)}\ (\kappa, \upsilon)\ t'\ \phi\right) + s\right) 0\ \varkappa
\end{aligned}
$$

$$
\mathrm{Compute}_{(\int)}\ (\kappa, \upsilon)\ t\ a\ \phi \triangleq \begin{cases} ev^{\eta}_{al}\ (\kappa, \upsilon)\ \phi\ (sub_{\mathbf{K}}\ (\kappa, \upsilon, t)\ a) & \text{if } a \geq 0 \\ \perp_{\mathbb{R}} & \text{otherwise} \end{cases}
$$

Synthesis of duration term: $\int^{a} \phi$

$$1_{\varphi(\kappa,\upsilon)} \quad :: (\mathbf{K} \times \Upsilon) \to \mathbb{R}_{\geq 0} \to \Phi^{3} \to \{0,1\}$$

$$1_{\varphi(\kappa,\upsilon)}\ (\kappa,\upsilon)\ t\ \phi \quad \triangleq \begin{cases} 1 & \text{if } \mathrm{Compute}_{\varphi}\ (\kappa,\upsilon,t)\ \phi = \mathsf{tt} \\ 0 & \text{otherwise} \end{cases}$$

$$ev^{\eta}_{al} \quad :: (\mathbf{K} \times \Upsilon) \to \Phi^{3} \to \mathbf{K} \to \mathbb{R}_{\geq 0}$$

$$ev^{\eta}_{al}\ (\kappa,\upsilon)\ \phi\ \varkappa \quad \triangleq fold\ \left(\lambda s, (p,(i,t')) \to t' \cdot \left(1_{\varphi(\kappa,\upsilon)}\ (\kappa,\upsilon)\ t'\ \phi\right) + s\right) 0\ \varkappa$$

$$\mathrm{Compute}_{(\int)}\ (\kappa,\upsilon)\ t\ a\ \phi \quad \triangleq \begin{cases} ev^{\eta}_{al}\ (\kappa,\upsilon)\ \phi\ (sub_{\mathbf{K}}\ (\kappa,\upsilon,t)\ a) & \text{if } a \geq 0 \\ \bot_{\mathbb{R}} & \text{otherwise} \end{cases}$$

Synthesis of Until operator: $\phi_1\ U_{<\gamma}\ \phi_2$

$$\text{Compute}_{(U_<)}\ m\ \gamma\ \phi_1\ \phi_2\ \triangleq \begin{cases} map^{\mathbb{B}_3}\ \left(ev^C_{al}\ m\ \gamma\ \phi_1\ \phi_2\ \left(sub_{\mathsf{K}}\ m\ \gamma\right)\right) & \text{if } \gamma \geq 0 \\ \mathsf{ff} & \text{otherwise} \end{cases}$$

Synthesis of Until operator: $\phi_1\ U_{<\gamma}\ \phi_2$

$$ev_{al}^{C} \qquad :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \mathbb{R}_{\geq 0} \to \Phi^3 \to \Phi^3 \to \mathbf{K} \to (\mathbb{B} \times \mathbf{B}_4)$$

$$ev_{al}^{C}\ (\kappa, \upsilon, t)\ \gamma\ \phi_1\ \phi_2\ \varkappa \quad \triangleq \left(d^{(\kappa)} \leq t + \gamma,\ \mathrm{ev}_{\mathrm{al}}^{\mathrm{fold}}\ (\kappa, \upsilon, t)\ \phi_1\ \phi_2\ \varkappa\right)$$

$$\mathrm{Compute}_{(U_<)}\ m\ \gamma\ \phi_1\ \phi_2 \triangleq \begin{cases} map^{\mathbb{B}_3}\ \left(ev_{al}^{C}\ m\ \gamma\ \phi_1\ \phi_2\ \left(sub_{\mathbf{K}}\ m\ \gamma\right)\right) & \text{if } \gamma \geq 0 \\ \mathsf{ff} & \text{otherwise} \end{cases}$$

Synthesis of Until operator: $\phi_1\ U_{<\gamma}\ \phi_2$

$$\begin{aligned}
&\mathrm{ev}_{\mathrm{al}}^{\mathrm{fold}} && :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \Phi^3 \to \Phi^3 \to \mathbf{K} \to \mathbf{B}_4 \\
&\mathrm{ev}_{\mathrm{al}}^{\mathrm{fold}}\ (\kappa, \upsilon, t)\ \phi_1\ \phi_2\ \varkappa && \triangleq \mathit{fold}\ \left(\lambda v\ (p, (i, t')) \to \mathrm{ev}_{\mathrm{al}}^{\mathrm{b}}\ (\kappa, \upsilon, t' - \epsilon)\ \phi_1\ \phi_2\ v\right) \mathfrak{r}\ \varkappa \\
&\mathit{ev}_{\mathit{al}}^{C} && :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \mathbb{R}_{\geq 0} \to \Phi^3 \to \Phi^3 \to \mathbf{K} \to (\mathbb{B} \times \mathbf{B}_4) \\
&\mathit{ev}_{\mathit{al}}^{C}\ (\kappa, \upsilon, t)\ \gamma\ \phi_1\ \phi_2\ \varkappa && \triangleq \left(d^{(\kappa)} \leq t + \gamma,\ \mathrm{ev}_{\mathrm{al}}^{\mathrm{fold}}\ (\kappa, \upsilon, t)\ \phi_1\ \phi_2\ \varkappa\right)
\end{aligned}$$

$$\mathrm{Compute}_{(U_<)}\ m\ \gamma\ \phi_1\ \phi_2 \triangleq \begin{cases} \mathit{map}^{\mathbb{B}_3}\ \left(\mathit{ev}_{\mathit{al}}^{C}\ m\ \gamma\ \phi_1\ \phi_2\ \left(\mathit{sub}_{\mathbf{K}}\ m\ \gamma\right)\right) & \text{if } \gamma \geq 0 \\ \mathsf{ff} & \text{otherwise} \end{cases}$$

Synthesis of Until operator: $\phi_1\ U_{<\gamma}\ \phi_2$

$$\mathrm{ev}^{\mathrm{b}}_{\mathrm{al}} \quad :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \mathbf{\Phi}^3 \to \mathbf{\Phi}^3 \to \mathbf{B}_4 \to \mathbf{B}_4$$

$$\mathrm{ev}^{\mathrm{b}}_{\mathrm{al}}\ m\ \phi_1\ \phi_2\ v \quad \triangleq \begin{cases} \mathrm{ev}^{\mathrm{i}}_{\mathrm{al}}\ (\mathrm{Compute}_{\varphi}\ m\ \phi_1)\ (\mathrm{Compute}_{\varphi}\ m\ \phi_2) & \text{if } v = \mathfrak{r} \\ v & \text{otherwise} \end{cases}$$

$$\mathrm{ev}^{\mathrm{fold}}_{\mathrm{al}} \quad :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \mathbf{\Phi}^3 \to \mathbf{\Phi}^3 \to \mathbf{K} \to \mathbf{B}_4$$

$$\mathrm{ev}^{\mathrm{fold}}_{\mathrm{al}}\ (\kappa, \upsilon, t)\ \phi_1\ \phi_2\ \varkappa \quad \triangleq \mathit{fold}\ \left(\lambda v\ (p, (i, t')) \to \mathrm{ev}^{\mathrm{b}}_{\mathrm{al}}\ (\kappa, \upsilon, t' - \epsilon)\ \phi_1\ \phi_2\ v\right) \mathfrak{r}\ \varkappa$$

$$ev^{C}_{al} \quad :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \mathbb{R}_{\geq 0} \to \mathbf{\Phi}^3 \to \mathbf{\Phi}^3 \to \mathbf{K} \to (\mathbb{B} \times \mathbf{B}_4)$$

$$ev^{C}_{al}\ (\kappa, \upsilon, t)\ \gamma\ \phi_1\ \phi_2\ \varkappa \quad \triangleq \left(d^{(\kappa)} \leq t + \gamma,\ \mathrm{ev}^{\mathrm{fold}}_{\mathrm{al}}\ (\kappa, \upsilon, t)\ \phi_1\ \phi_2\ \varkappa\right)$$

$$\mathrm{Compute}_{(U_<)}\ m\ \gamma\ \phi_1\ \phi_2 \quad \triangleq \begin{cases} map^{\mathbb{B}_3}\ \left(ev^{C}_{al}\ m\ \gamma\ \phi_1\ \phi_2\ (sub_{\mathbf{K}}\ m\ \gamma)\right) & \text{if } \gamma \geq 0 \\ \mathsf{ff} & \text{otherwise} \end{cases}$$

Synthesis of Until operator: $\phi_1\ U_{<\gamma}\ \phi_2$

$$\mathrm{ev}^{\mathrm{i}}_{\mathrm{al}} \quad :: \mathbb{B}_3 \to \mathbb{B}_3 \to \mathbf{B}_4$$

$$\mathrm{ev}^{\mathrm{i}}_{\mathrm{al}}\ b_1\ b_2 \triangleq \begin{cases} map^{\mathbf{B_4}}\ b_2 & \text{if } b_2 \neq \mathrm{ff} \\ map^{\mathbf{B_4}}\ b_1 & \text{if } b_1 \neq \mathrm{tt} \text{ and } b_2 = \mathrm{ff} \\ \mathfrak{r} & \text{otherwise} \end{cases}$$

$$\mathrm{ev}^{\mathrm{b}}_{\mathrm{al}} \quad :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \Phi^3 \to \Phi^3 \to \mathbf{B}_4 \to \mathbf{B}_4$$

$$\mathrm{ev}^{\mathrm{b}}_{\mathrm{al}}\ m\ \phi_1\ \phi_2\ v \triangleq \begin{cases} \mathrm{ev}^{\mathrm{i}}_{\mathrm{al}}\ (\mathrm{Compute}_{\varphi}\ m\ \phi_1)\ (\mathrm{Compute}_{\varphi}\ m\ \phi_2) & \text{if } v = \mathfrak{r} \\ v & \text{otherwise} \end{cases}$$

$$\mathrm{ev}^{\mathrm{fold}}_{\mathrm{al}} \quad :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \Phi^3 \to \Phi^3 \to \mathbf{K} \to \mathbf{B}_4$$

$$\mathrm{ev}^{\mathrm{fold}}_{\mathrm{al}}\ (\kappa, \upsilon, t)\ \phi_1\ \phi_2\ \varkappa \triangleq fold\ \left(\lambda v\ (p, (i, t')) \to \mathrm{ev}^{\mathrm{b}}_{\mathrm{al}}\ (\kappa, \upsilon, t' - \epsilon)\ \phi_1\ \phi_2\ v\right)\ \mathfrak{r}\ \varkappa$$

$$ev^{C}_{al} \quad :: (\mathbf{K} \times \Upsilon \times \mathbb{R}_{\geq 0}) \to \mathbb{R}_{\geq 0} \to \Phi^3 \to \Phi^3 \to \mathbf{K} \to (\mathbb{B} \times \mathbf{B}_4)$$

$$ev^{C}_{al}\ (\kappa, \upsilon, t)\ \gamma\ \phi_1\ \phi_2\ \varkappa \triangleq \left(d^{(\kappa)} \leq t + \gamma,\ \mathrm{ev}^{\mathrm{fold}}_{\mathrm{al}}\ (\kappa, \upsilon, t)\ \phi_1\ \phi_2\ \varkappa\right)$$

$$\mathrm{Compute}_{(U_<)}\ m\ \gamma\ \phi_1\ \phi_2 \triangleq \begin{cases} map^{\mathbb{B}_3}\ \left(ev^{C}_{al}\ m\ \gamma\ \phi_1\ \phi_2\ (sub_{\mathbf{K}}\ m\ \gamma)\right) & \text{if } \gamma \geq 0 \\ \mathrm{ff} & \text{otherwise} \end{cases}$$

Settling decisions about our synthesis mechanism

Lets introduce other available and common mechanisms.

What we get by using automata ?

A lot of theory and fundamental properties. But, **could such approach help us on runtime verification of duration formulas?** Maybe not.

What we gain using automata with stop-watches?

We know that the **reachability problem** is **undecidable** when more than one clock is used.

- Since we are dealing with explicit time, **automata based synthesization is irrelevant when compared with the exploration of inductive structures**. Well, a very strong and unproved sentence.

Settling decisions about our synthesis mechanism

Lets introduce other available and common mechanisms.

What we get by using automata ?

A lot of theory and fundamental properties. But, **could such approach help us on runtime verification of duration formulas?** Maybe not.

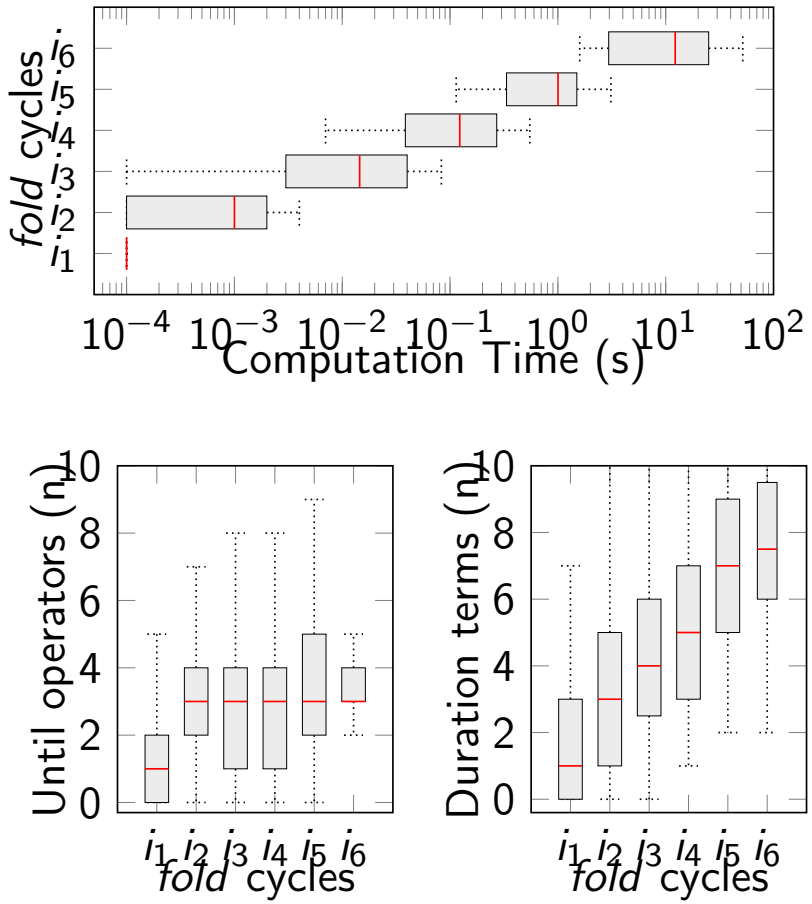
What we gain using automata with stop-watches?

We know that the **reachability problem** is **undecidable** when more than one clock is used.

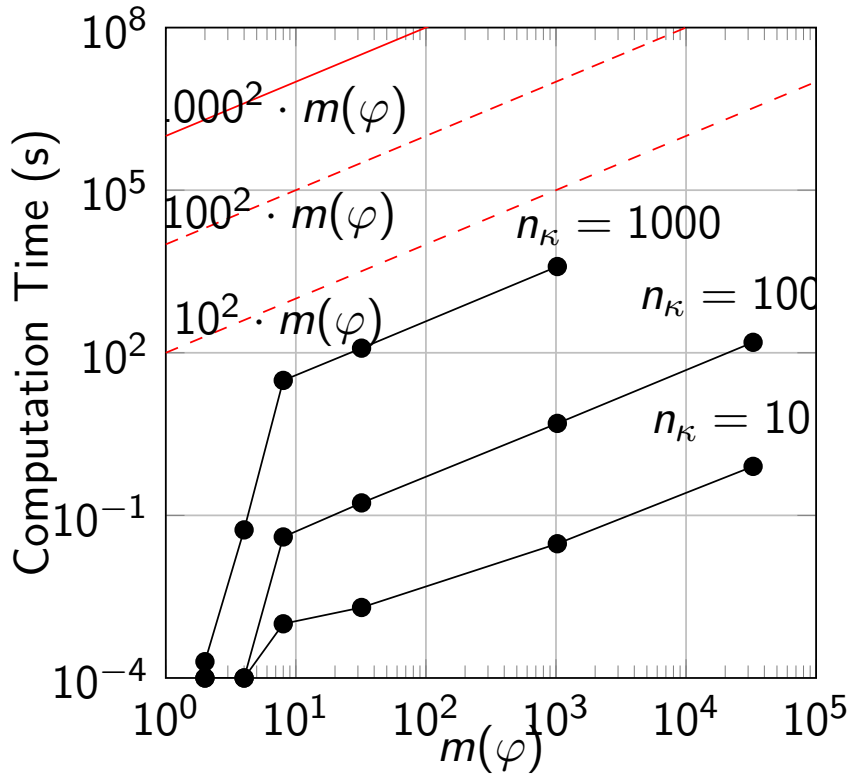
- Since we are dealing with explicit time, **automata based synthesization is irrelevant when compared with the exploration of inductive structures**. Well, a very strong and unproved sentence.

*** AND NOW SIMULATIONS ***

Simulated results of the RMTL-$\int$ evaluation algorithm



(a) computation time vs. execution cycles of fold functions, $m(\varphi) = 2^5 - 1$ and $n_\kappa = 1000$



(b) computation time vs. formula size constructed with nested Until operators

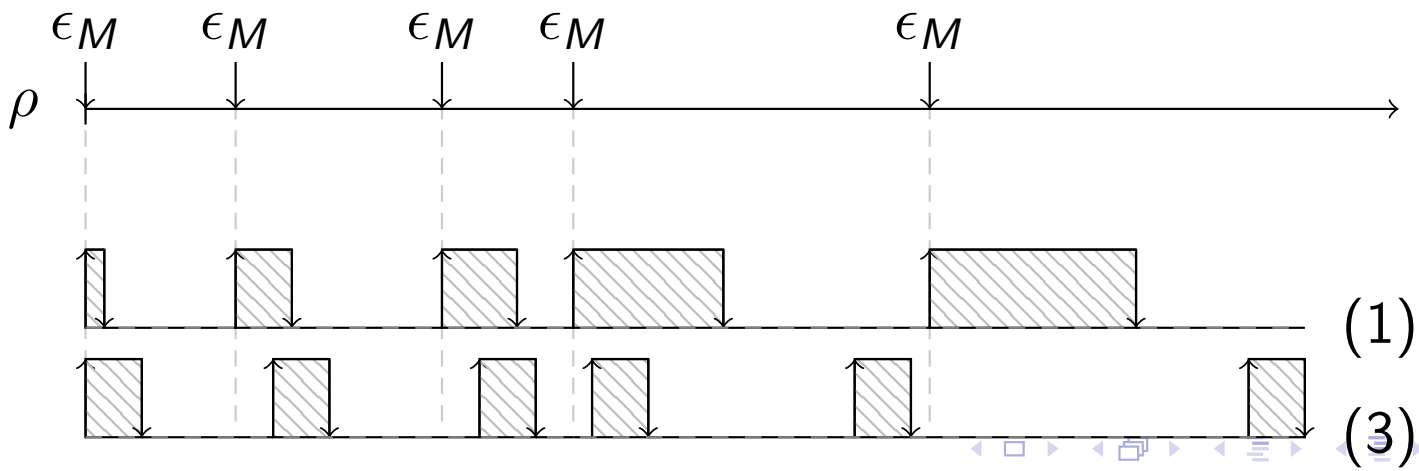
Figure: Experimental validation of the complexity results provided in [6]

Talk Outline

Unstable conditions for monitors

Pre-defined conditions in RML4Ada framework

1. step-bounded condition – the execution of the monitor ends when n iterations have been processed or when events have not arrived (dynamic time execution);
2. time-bounded condition – the execution of the monitor is bounded by t time units, exiting if no events occur (dynamic time execution);
3. symbol-based condition – the execution ends when one or more symbols of the path are consumed, and the monitor sleeps until a new symbol arrival (dynamic time execution);



Stable conditions for monitors

Unstable monitors are not enough to ensure safe monitoring. For that we require an hierarchy of well-behaved monitors and well-dependent monitors. **Correctness by construction.**

But, what this means?

We need to make the assumption that the monitor is **time constant**. Each branch of a control flow graph of a monitor has to run exactly the same instructions or at least the same number of instructions (note that the latter is more weak).

Based on that we can trust over the verdicts of such monitor, and employ unstable monitors to perform more dynamic tasks.
We need to assume that monitors are always interfering as a constant measure of time. Can we could call it stationary ?

Talk Outline

Talk Outline

PixHawk platform

- Applicability of the developed tools for a real application using a real embedded ARM Cortex-M4 processor.
- Lightweight Drones are currently unsafe in terms of time-space isolation. Our purpose is to apply the runtime verification to increase dependability of such well spread air vehicles.

Talk Outline

RTEML is for embedded systems I

RTEML is a modification of the current in house multi-purpose approach and library [7] that is purely imperative programming oriented.

Issues that we identify for embedded systems

- assigning a buffer for each *type of event* is memory consuming. A buffer of 100 elements is not the same as a set of ten buffers of size 10 (related to the approach);
- event overwrite is not safe since can be unrecoverable at some point (related to the approach);
- only works if monitors have less priority (related to the library level);
- lock-free mechanism is supported by Boost library (related to the library level);
- wait-free is not provided (related to the library level);

RTEML is for embedded systems II

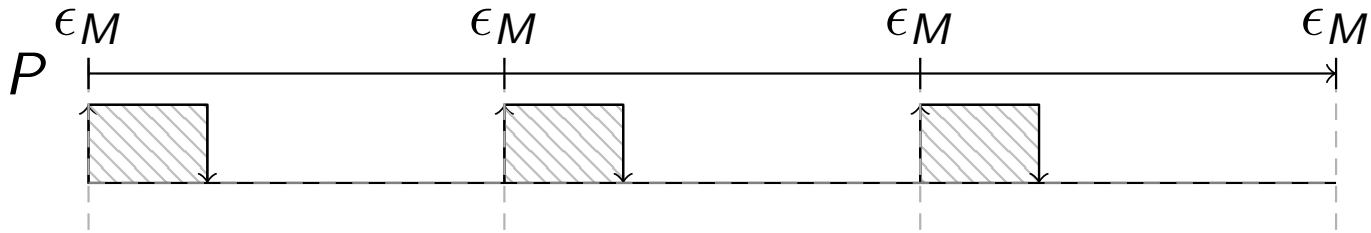
RTEML usage regards

- is lock- and wait-free;
- does not overwrite events by default;
- due to the synthesis tool it manages the buffers of monitors by clusters when synthesis of RMTL-$\int_3$ is used;
- only works for ARM architectures but can be further extended;

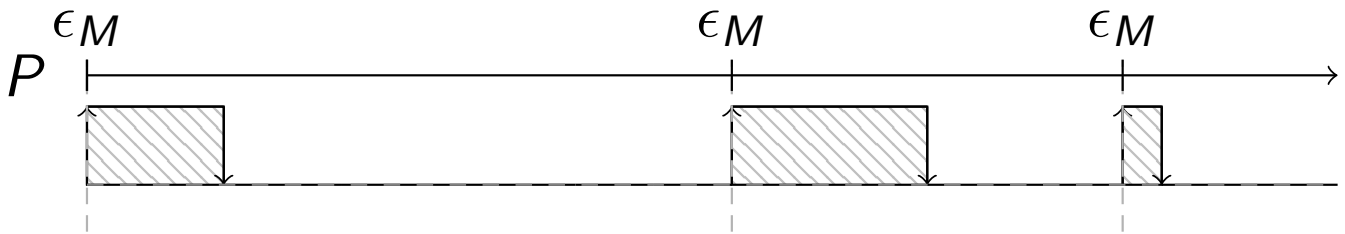
Modes of execution for RTEML

Modes of operation

- Time-Triggered mode – executes as a periodic task;



- Event-Based mode – executes as a sporadic task (each inter-arrival time shall be supplied before the execution and statically checked);



- Hybrid mode – mixture both previous modes;

Talk Outline

From RMTL-$\int_3$ to C++11

Thanks to C++11 lambda functions.
Thanks to Ocaml programming language. Lets see the *rmtld3synthcpp* tool in more detail...

DEMO

The End...

Thank you for watching our presentation.
Please send any comment to anmap@isep.ipp.pt.

Publications I

Logic-based Schedulability Analysis for Compositional Hard Real-Time Embedded Systems.
In *Proceedings of the 6th International Workshop on Compositional Theory and Technology for Real-Time Embedded Systems*, CRTS '13, 2013.

A Compositional Monitoring Framework for Hard Real-Time Systems.
In *Proceedings of the 6th NASA Formal Methods Symposium*, pages 16–30, 2014.

Implementing Ada Task Types in AVR-Ada.
Ada User Journal, 35(3):194–203, 2014.

Publications II

Towards a Runtime Verification Framework for the Ada Programming Language.
In *Proceedings of the 19th Ada-Europe International Conference on Reliable Software Technologies*, pages 58–73, 2014.

Logic-based schedulability analysis for compositional hard real-time embedded systems.
SIGBED Rev., 12(1):56–64, March 2015.

Monitoring for a decidable fragment of MTLD.
In *Proceedings of the 15th International Conference on Runtime Verification*, September 2015.

Publications III


Geoffrey Nelissen, David Pereira, and LuísMiguel Pinho.
A novel run-time monitoring architecture for safe and efficient inline monitoring.
In Juan Antonio de la Puente and Tullio Vardanega, editors, *Reliable Software Technologies – Ada-Europe 2015*, volume 9111 of *Lecture Notes in Computer Science*, pages 66–82. Springer International Publishing, 2015.