FROM REAL-TIME LOGIC TO TIMED AUTOMATA

DEJAN NIČKOVIĆ – AIT AUSTRIAN INSTITUTE OF TECHNOLOGY

Joint work with Thomas Ferrère, Oded Maler and Amir Pnueli

MOVEP'20
MOTIVATION

- CPS often have **hard real-time** constraints

- Real-time specification language interpreted over dense time
  - Metric Interval Temporal Logic (MITL)

- Need for an RV approach to evaluate such specs
  - Timed automata (TA) as real-time observers

- How to translate MITL to TA?
  - **Monolithic** tableau construction in [AFH96]

Our contribution

- Novel **compositional** translation
  - **Temporal testers** instead of acceptor automata
  - Facilitates adding extensions
  - Handles past operators for free
LINEAR TEMPORAL LOGIC (LTL)

Syntax

\[ S := p \mid \neg S \mid S_1 \text{ or } S_2 \mid \text{next } S \mid S_1 \text{ until } S_2 \]

<table>
<thead>
<tr>
<th>True</th>
<th>\equiv S \text{ or not } S</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>\equiv \neg \text{true}</td>
</tr>
<tr>
<td>(S_1 \text{ and } S_2)</td>
<td>\equiv \neg (\neg S_1 \text{ or } \neg S_2)</td>
</tr>
<tr>
<td>(S_1 \text{ implies } S_2)</td>
<td>\equiv \neg S_1 \text{ or } S_2</td>
</tr>
<tr>
<td>(\text{eventually } S)</td>
<td>\equiv \text{true until } S</td>
</tr>
<tr>
<td>(\text{always } S)</td>
<td>\equiv \neg \text{eventually not } S</td>
</tr>
</tbody>
</table>

Semantics

- \(p\)
- \(\text{next } p\)
- \(\text{eventually } p\)
- \(\text{always } p\)
- \(p \text{ until } q\)
What are temporal testers?

- **Non-deterministic** sequential transducers
  - Inputs and outputs

- Realize a temporal logic function
  - Tester $T_S$ for specification $S$
  - $T_S(w, t) = 1 \iff (w, t) \models S$

- Modular construction
  - Temporal testers defined for basic temporal operators
  - Arbitrary formula = composition
  - Example: $always(req \implies eventually gnt)$

\begin{align*}
\neg p / \neg u & \quad p / \neg u \\
\neg p / u & \quad p / u
\end{align*}
COMPOSITION OF TEMPORAL TESTERS

- Composition of Transducers – Synchronization on I/O
TEMPORAL TESTERS VS. ACCEPTORS

- Acceptor $A_S$
  - Accepts trace $w$ iff $(w, 0) \models S$
- Temporal Tester $T_S$
  - Checks if $(w, t) \models S$ for all $t \geq 0$
  - Acceptor for every suffix of $w$

- Given acceptors $A_{S_1}$ and $A_{S_2}$ for $S_1$ and $S_2$
  - No simple recipe to compose them to get acceptor for $S_1 until S_2$
- Given temporal testers $T_{S_1}$ and $T_{S_2}$ for $S_1$ and $S_2$
  - Composition yields tester for $S_1 until S_2$

<table>
<thead>
<tr>
<th>time</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$S_2$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$A_{S_1}$</td>
<td>1</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>$A_{S_2}$</td>
<td>0</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>$A_{S_1 until S_2}$</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>$T_{S_1}$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$T_{S_2}$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$T_{S_1 until S_2}$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
TEMPORAL TESTER FOR NEXT

- Non-causal shift register

\[ u = \text{next} \ p \]
TEMPORAL TESTER FOR NEXT

- Non-causal shift register

\[ u = \text{next } p \]
Non-causal shift register

\[ u = \text{next } p \]
TEMPORAL TESTER FOR NEXT

- Non-causal shift register

\[ u = \text{next } p \]
TEMPORAL TESTER FOR NEXT

- Non-causal shift register

\[ u = \text{next } p \]
TEMPORAL TESTER FOR NEXT

- Non-causal shift register

\[ u = \text{next } p \]
Temporal logic over **continuous-time** signals

**Syntax**

\[ S := p \mid \text{not} \ S \mid S_1 \text{ or } S_2 \mid S_1 \text{ until}_I S_2 \]

- \( I \) is a **non-punctual** interval
- Other Boolean and temporal operators derived as expected
- We can also consider MITL with past operators

**Satisfiability** of MITL is **EXPSPACE-complete**.

**Semantics**

\( \begin{align*}
  p & \quad \text{eventually}_{[0,1]} p \\
  q & \quad \text{always}_{[1,3]} p \\
  p \text{ until}_{[0,1]} q
\end{align*} \)
Temporal logic over **continuous-time** signals

- **Syntax**
  
  \[ S := p \mid \text{not } S \mid S_1 \text{ or } S_2 \mid S_1 \text{ until}_I S_2 \]

- \( I \) is a **non-punctual** interval

- Other Boolean and temporal operators derived as expected

- We can also consider MITL with past operators

- **Satisfiability** of MITL is **EXPSPACE-complete**.

---

**Semantics**

- \( p \) eventually \([0,1] p\)

- \( p \) always \([1,3] p\)

- \( p \) until \([0,1] q\)
Temporal logic over **continuous-time** signals

**Syntax**

\[ S := p \mid not S \mid S_1 \ or \ S_2 \mid S_1 \ until_I \ S_2 \]

- \( I \) is a **non-punctual** interval
- Other Boolean and temporal operators derived as expected
- We can also consider MITL with past operators

**Satisfiability** of MITL is **EXPSPACE-complete**.

**Semantics**

![Graphical representation of MITL operators](image-url)
Temporal logic over continuous-time signals

**Syntax**

\[ S := p \mid \text{not } S \mid S_1 \text{ or } S_2 \mid S_1 \text{ until}_I S_2 \]

- \( I \) is a non-punctual interval
- Other Boolean and temporal operators derived as expected
- We can also consider MITL with past operators

**Satisfiability** of MITL is EXPSPACE-complete.
SOME ADDITIONAL NOTES ON MITL

**Signals**

- Multi-dimensional Boolean signals
  - $w : R_+ \to B^n$
- Alternating sequence of **singular points** and open intervals

**Semantics**

- Strict definition of until

$$ (w, t) \models S_1 \text{until}_1 S_2 \iff \exists t' \in t \oplus I \text{ s.t. } (w, t') \models S_2 \text{ and } \forall t'' \in (t, t'), (w, t'') \models S_1 $$

- $S_1 \text{ until } S_2 = S_1 \text{ until}_{(0, \infty)} S_2$
- next $S = S \text{ until } S$
- TA with inputs and outputs
- Auxiliary **clocks**
  - Location **invariants**
  - Transition **guards** and **resets**
- Labels on both transitions and locations
- Generalized Büchi conditions on locations

```
x := 0
\neg p/\neg u
\neg p/u
x > 0
\neg p/\neg u
\neg p/\neg u
p/u
x < 2
p/u
\neg p/u
```

**clock guard**

**clock reset**

**clock invariant**
FROM MITL TO AUTOMATA – FORMULA SIMPLIFICATION

- We only need a temporal tester for $until_1$!
- Building temporal testers for $until_1$ is hard
  - Simplify specifications
- Eliminate $until_1$
  - $p \ until_{(a,b)} q = p \ until_{(a,\infty)} q$ and $\ eventually_{(a,b)} q$
  - $p \ until_{(c,\infty)} q = \ always_{(0,c]}(p \ and p \ until \ q)$

$$p \ until_{(a,b)} q = \ always_{(0,a)} p \ \land \ \ always_{(0,a]}(p \ until \ q) \ \land \ \ eventually_{(a,b)} q$$
FROM MITL TO AUTOMATA – FORMULA SIMPLIFICATION

- We only need a temporal tester for $until_1$!
- Building temporal testers for $until_1$ is hard
  - Simplify specifications
- Eliminate $until_1$
  - $p until_{(a,b)}q = p until_{(a,\infty)}q \land eventually_{(a,b)}q$
  - $p until_{(c,\infty)}q = always_{[0,c]}(p \land p until q)$

\[
p until_{(a,b)}q = always_{(0,a)}p \land always_{(0,a)}(p until q) \land eventually_{(a,b)}q
\]
FROM MITL TO AUTOMATA – FORMULA SIMPLIFICATION

- We only need a temporal tester for $until_I$!
- Building temporal testers for $until_I$ is hard
  - Simplify specifications
- Eliminate $until_I$
  - $p \ until_{(a,b)} q = p \ until_{(a,\infty)} q$ and $eventually_{(a,b)} q$
  - $p \ until_{(c,\infty)} q = always_{(0,c]} (p \ and \ p \ until \ q)$

$$p \ until_{(a,b)} q = always_{(0,a)} p \land always_{(0,a]} (p \ until \ q) \land eventually_{(a,b)} q$$
We only need a temporal tester for $\text{until}_I$!

Building temporal testers for $\text{until}_I$ is hard

- Simplify specifications

Eliminate $\text{until}_I$

- $p \text{ until}_{(a,b)} q = p \text{ until}_{(a,\infty)} q$ and $\text{eventually}_{(a,b)} q$
- $p \text{ until}_{(c,\infty)} q = \text{always}_{[0,c]}(p \text{ and } p \text{ until } q)$

Eliminate $\text{eventually}$ with positive lower bound

- $\text{eventually}_{(a+c,b+c)} p = \text{eventually}_{(0,c)} \text{ always}_{(0,c)} \text{ eventually}_{(a,b)} p$
- $\text{eventually}_{(0,a)} p = \text{eventually}_{(0,a)}$ or $(\text{next always}_{(0,a)}$ and $(\text{not } p)$ until $p)$

It is sufficient to build temporal testers for $\text{until}$ and $\text{eventually}_{(0,a)}$
**Intuition**

- Assume
  - Signals with left-closed right-open intervals only
  - Non-strict semantics of until
    - $p \text{ until } q = q \text{ or } (p \text{ and } p \text{ until } q)$

$$u = p \text{ until } q$$

$$\begin{array}{c}
p \quad \text{ until } q \quad u \\
\neg p \quad \neg u \\
p \land q \quad \neg u \\
p \land \neg q \quad \neg u \\
p \land \neg q \quad \neg u \\
p \land q / u \\
\end{array}$$
**Intuition**

- Assume
  - Signals with left-closed right-open intervals only
  - Non-strict semantics of until

\[ u = p \text{ until } q \]
Intuition

- Assume
  - Signals with left-closed right-open intervals only
  - Non-strict semantics of until
    - \( p \text{ until } q = (p \text{ and } q) \text{ or } (p \text{ and } p \text{ until } q) \)

\[ u = p \text{ until } q \]

\[ \neg p/\neg u \]

\[ p \land q/u \]

\[ p \land \neg q/\neg u \]

\[ p \land \neg q/u \]
**TEMPORAL TESTER FOR UNTIL**

**Intuition**

- Assume
  - Signals with left-closed right-open intervals only
  - Non-strict semantics of until
    - $p \text{ until } q = q$ or ($p$ and $p \text{ until } q$)

```
p
q
u  
```

```
\begin{align*}
\neg p & \text{ until } q = q \\
\neg q & \text{ until } u = \neg u
\end{align*}
```

```
u = p \text{ until } q
```
Intuition

- Assume
  - Signals with left-closed right-open intervals only
  - Non-strict semantics of until
    - $p \text{ until } q = q$ or ($p$ and $p \text{ until } q$)

\[
\begin{align*}
\neg p / \neg u & \quad p \land q / u \\
\neg p \land q / u & \quad p \land \neg q / u
\end{align*}
\]
TEMPORAL TESTER FOR UNTIL

**Intuition**

- **Assume**
  - Signals with left-closed right-open intervals only
  - Non-strict semantics of until
    - \( p \) until \( q = q \) or \( (p \) and \( p \) until \( q) \)

\[
\begin{align*}
\neg p / \neg u & \Rightarrow p \land q / u \\
p \land \neg q / \neg u & \Rightarrow p \land \neg q / u
\end{align*}
\]

\[
u = p \text{ until } q
\]
TEMPORAL TESTER FOR UNTIL

**Intuition**

- **Assume**
  - Signals with left-closed right-open intervals only
  - Non-strict semantics of until
    - \( p \text{ until } q = q \text{ or } (p \text{ and } p \text{ until } q) \)

\[
\begin{align*}
    u &= p \text{ until } q \\
    \neg p / \neg u &\quad \rightarrow \quad p \wedge q / u \\
    p \wedge \neg q / \neg u &\quad \rightarrow \quad p \wedge \neg q / u
\end{align*}
\]
**Intuition**

- **Assume**
  - Signals with left-closed right-open intervals only
  - Non-strict semantics of until
    - $p \text{ until } q = q$ or $(p$ and $p \text{ until } q)$

\[ u = p \text{ until } q \]

\[ \begin{align*}
\neg p/\neg u & \quad \text{ or } \quad p \land q/u \\
\neg p/\neg u & \quad \text{ or } \quad p \land q/u \\
p \land \neg q/\neg u & \quad \text{ or } \quad p \land \neg q/\neg u
\end{align*} \]
Full construction

- $p \text{ until } q$ is right-continuous
  - $(w, t) \models p \text{ until } q \rightarrow \exists t' > t, \forall t'' \in (t', t) \ (w, t'') \models p \text{ until } q$
  - $u = u(t_0) \cdot u(t_0, t_1) \cdot u(t_1) \cdot u(t_1, t_2) \cdots$
  - $u(t_i) = u(t_i, t_{i+1}), \forall i \geq 0$

<table>
<thead>
<tr>
<th>$w(t_i)$</th>
<th>$w(t_i, t_{i+1})$</th>
<th>$u(t_i, t_{i+1})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>$p \land q$</td>
<td>1</td>
</tr>
<tr>
<td>*</td>
<td>$\neg p$</td>
<td>0</td>
</tr>
<tr>
<td>$q$</td>
<td>$p \land \neg q$</td>
<td>1</td>
</tr>
<tr>
<td>$\neg p \land \neg q$</td>
<td>$p \land \neg q$</td>
<td>0</td>
</tr>
<tr>
<td>$p \land \neg q$</td>
<td>$p \land \neg q$</td>
<td>$u(t_i)$</td>
</tr>
</tbody>
</table>
Intuition

- Assume
  - Signals with left-closed right-open intervals only
  - Formula of the form $\text{eventually}_{[0,a]} p$

$p$

$u$

\[ u = \text{eventually}_{[0,a]} p \]

\[
\begin{align*}
\neg p/u & \quad \text{c := 0} \\
\neg p/u & \quad \text{c < a} \\
\neg \neg p/u & \quad \text{c := 0} \\
\neg \neg p/u & \quad \text{c \leq a}
\end{align*}
\]
Intuition

- Assume
  - Signals with left-closed right-open intervals only
  - Formula of the form \( \text{eventually}_{[0,a]} p \)

\[ u = \text{eventually}_{[0,a]} p \]
**Intuition**

- **Assume**
  - Signals with left-closed right-open intervals only
  - Formula of the form \( \text{eventually}_{[0,a]} p \)

\[
\text{eventually}_{[0,a]} p = \neg \neg p/\neg u \quad \text{with } \neg p/u \quad c < a
\]

\[
c := 0 \quad \text{and } c = a
\]
Intuition

- Assume
  - Signals with left-closed right-open intervals only
  - Formula of the form \( eventually_{[0,a]} p \)

\[
\begin{align*}
  u &= eventually_{[0,a]} p \\
  p/u &\xrightarrow{c := 0} \neg p/u \\
  \neg p/\neg u &\xrightarrow{c := 0} \neg p/u \\
  \neg p/\neg u &\xrightarrow{c = a} \neg p/u \\
  \neg p/\neg u &\xrightarrow{c = a} \neg p/\neg u
\end{align*}
\]
**Intuition**

- **Assume**
  - Signals with left-closed right-open intervals only
  - Formula of the form $\text{eventually}_{[0,a]} p$

(u = eventually_{[0,a]} p)

\[
\begin{align*}
p / u & \\
\neg p / \neg u & \\
\quad c := 0 & \quad c < a \\
\quad c = a & \\
\quad \neg p / u & \\
\quad c := 0 & \quad c \leq a
\end{align*}
\]
TEMPORAL TESTER FOR EVENTUALLY

Intuition

- Assume
  - Signals with left-closed right-open intervals only
  - Formula of the form $\text{eventually}_{[0,a]} p$

\[
\begin{align*}
  u &= \text{eventually}_{[0,a]} p \\
  p/u &
  \xrightarrow[c := 0]{c = a} \\
  \neg p/u &
  \xrightarrow[c := 0]{c \leq a} \\
  \neg p/\neg u &\xrightarrow{c := 0}
\end{align*}
\]
**TEMPORAL TESTER FOR EVENTUALLY**

**Intuition**

- Assume
  - Signals with left-closed right-open intervals only
  - Formula of the form $\text{eventually}_{[0,a]} p$

![Diagram](attachment:image.png)

$u = \text{eventually}_{[0,a]} p$

- $c := 0$
- $c < a$
- $c = a$
- $c \leq a$
Assume
- Signals with left-closed right-open intervals only
- Formula of the form \( \text{eventually}_{[0,a]} p \)

\[
\begin{align*}
\neg p/u & \leftarrow c := 0 \\
\neg p/u & \leftarrow c < a \\
p/u & \leftarrow c = a \\
\neg p/\neg u & \leftarrow c := 0 \\
\neg p/\neg u & \leftarrow c \leq a \\
\end{align*}
\]
TEMPORAL TESTER FOR EVENTUALLY

Full construction

- Signals with arbitrary intervals
- Formula of the form $\text{eventually}_{(0,a)} p$

\[ u = \text{eventually}_{(0,a)} p \]
CONCLUSIONS

- Details
- MITL can be fully expressed with two (simpler than $U_I$) temporal operators
  - $U$ and $F_{(0,a)}$
- We provide a construction of timed testers for these operators
  - Straight-forward to understand and implement
  - At most 4 locations and 1 clock per tester
- Network of communicating testers, derived from the structure of an MITL property, yields an equivalent timed automaton
- Practical applications
  - Model checking, runtime verification, coverage-based testing, …
THANK YOU FOR YOUR ATTENTION