Visibly Linear Dynamic Logic

Joint work with Alexander Weinert (Saarland University)

Martin Zimmermann

Saarland University

September 8th, 2016

Highlights Conference, Brussels, Belgium
The Everlasting Quest for Expressiveness

- **LTL:** “Every request $q$ is eventually answered by a response $p$”

  \[ G(q \rightarrow F p) \]
The Everlasting Quest for Expressiveness

- **LTL**: “Every request \( q \) is eventually answered by a response \( p \)”
  
  \[ G(q \rightarrow F p) \]

- **LDL**: “Every request \( q \) is eventually answered by a response \( p \) after an even number of steps”
  
  \[ [true^*](q \rightarrow ((true \cdot true)^* \langle p \rangle) \]

- **VLDL**: “Every request \( q \) is eventually answered by a response \( p \) and there are never more responses than requests”

  This can be expressed using pushdown automata/context-free grammars in the guards.
The Everlasting Quest for Expressiveness

- **LTL:** “Every request \( q \) is eventually answered by a response \( p \)”
  \[ G(q \rightarrow F p) \]

- **LDL:** “Every request \( q \) is eventually answered by a response \( p \) after an even number of steps”
  \[ \text{[true}^*]\](q \rightarrow ((\text{true} \cdot \text{true})^*)p) \]

- **VLDL:** “Every request \( q \) is eventually answered by a response \( p \) and there are never more responses than requests”
The Everlasting Quest for Expressiveness

- **LTL**: “Every request $q$ is eventually answered by a response $p$”
  \[ G(q \rightarrow F p) \]

- **LDL**: “Every request $q$ is eventually answered by a response $p$ after an even number of steps”
  \[ [\text{true}^*](q \rightarrow \langle (\text{true} \cdot \text{true})^* \rangle p) \]

- **VLDL**: “Every request $q$ is eventually answered by a response $p$ and there are never more responses than requests”
  This can be expressed using pushdown automata/context-free grammars in the guards.
Partition input alphabet $\Sigma$ into $\Sigma_c$ (calls), $\Sigma_r$ (returns), and $\Sigma_\ell$ (local actions).

A visibly pushdown automaton (VPA) has to
- push when processing a call,
- pop when processing a return while the stack is non-empty (otherwise stack is unchanged), and
- leave the stack unchanged when processing a local action.

Stack height determined by input word $\Rightarrow$ closure under union, intersection, and complement.
Visibly Pushdown Automata

Partition input alphabet $\Sigma$ into $\Sigma_c$ (calls), $\Sigma_r$ (returns), and $\Sigma_\ell$ (local actions).

A visibly pushdown automaton (VPA) has to

- push when processing a call,
- pop when processing a return while the stack is non-empty (otherwise stack is unchanged), and
- leave the stack unchanged when processing a local action.

Stack height determined by input word $\Rightarrow$ closure under union, intersection, and complement.

Examples:

- $a^n b^n$ is a VPL, if $a$ is a call and $b$ a return.
- $ww^R$ is not a VPL.
Visibly Linear Dynamic Logic (VLDDL)

Syntax

\[ \varphi ::= p \mid \neg \varphi \mid \varphi \land \varphi \mid \varphi \lor \varphi \mid \langle A \rangle \varphi \mid [A] \varphi \]

where \( p \in P \) ranges over atomic propositions and \( A \) ranges over VPA’s. All VPA’s have the same partition of \( 2^P \) into calls, returns, and local actions.
Visibly Linear Dynamic Logic (VLDL)

Syntax

\[ \varphi ::= p \mid \neg \varphi \mid \varphi \land \varphi \mid \varphi \lor \varphi \mid \langle A \rangle \varphi \mid [A] \varphi \]

where \( p \in P \) ranges over atomic propositions and \( A \) ranges over VPA’s. All VPA’s have the same partition of \( 2^P \) into calls, returns, and local actions.

Semantics

- \( w \models \langle A \rangle \varphi \) if there exists an \( n \) such that \( w_0 \cdots w_n \) is accepted by \( A \) and \( w_n w_{n+1} w_{n+2} \cdots \models \varphi \).

- \( w \models [A] \varphi \) if for every \( n \) s.t. \( w_0 \cdots w_n \) is accepted by \( A \) we have \( w_n w_{n+1} w_{n+2} \cdots \models \varphi \).
“Every request $q$ is eventually answered by a response $p$ and there are never more responses than requests”:

$$[\mathcal{A}_{\text{true}}](q \rightarrow \langle \mathcal{A}_{\text{true}} \rangle p) \land [\mathcal{A}]\text{false}$$

where

- $\mathcal{A}_{\text{true}}$ accepts every input, and
- $\mathcal{A}$ accepts every input with more responses than requests.

Both languages are visibly pushdown, if

- $\{q\}$ is a call,
- $\{p\}$ is a return, and
- $\emptyset$ and $\{p, q\}$ are local actions.
Lemma

VLDL and non-deterministic $\omega$-VPA are expressively equivalent.
Expressiveness

Lemma

VLDL and non-deterministic \( \omega \)-VPA are expressively equivalent.

Proof Idea

\[
\text{VLDL}
\]

\[
\text{non-deterministic} \\
\omega \text{-VPA}
\]
Expressiveness

**Lemma**

*VLDL and non-deterministic ω-VPA are expressively equivalent.*

**Proof Idea**

![Diagram showing the relationship between VLDL, Deterministic Stair Automata, and non-deterministic ω-VPA, with a complexity of O(2^n).]
Expressiveness

Lemma

VLDL and non-deterministic $\omega$-VPA are expressively equivalent.

Proof Idea

\[ O(2^n) \]

Deterministic Stair Automata

\[ O(n^2) \]

VLDL

non-deterministic $\omega$-VPA

[Bozelli '07]

\[ O(2^n) \]

[LMS '04]
Expressiveness

Lemma

VLDL and non-deterministic ω-VPA are expressively equivalent.

Proof Idea

Deterministic Stair Automata

[O(2^n)]

non-deterministic ω-VPA

VLDL

O(n^2)

1-way Alternating Jumping Automata

O(n^2)
Expressiveness

Lemma

VLDL and non-deterministic $\omega$-VPA are expressively equivalent.

Proof Idea

$O(2^n)$

Deterministic Stair Automata

VLDL

$O(n^2)$

1-way Alternating Jumping Automata

$O(n^2)$

non-deterministic $\omega$-VPA

$O(2^n)$

[Bozelli '07]

[LMS '04]
“If $p$ holds true immediately after entering module $m$, it shall hold immediately after the corresponding return from $m$ as well”
“If $p$ holds true immediately after entering module $m$, it shall hold immediately after the corresponding return from $m$ as well”

**VLDL:**

$$[A_c](p \rightarrow \langle A_r \rangle p)$$

with

$$\begin{align*}
\Sigma_r, \uparrow A & \quad \Sigma_c, \downarrow A \quad \Sigma_\ell, \rightarrow \\
\Sigma_c, \downarrow A & \quad \Sigma_\ell, \rightarrow
\end{align*}$$

$$\begin{align*}
\Sigma_r, \uparrow A & \quad \Sigma_c, \downarrow A \quad \Sigma_\ell, \rightarrow \\
\Sigma_c, \downarrow A & \quad \Sigma_\ell, \rightarrow
\end{align*}$$

$A_c$

$A_r$
“If $p$ holds true immediately after entering module $m$, it shall hold immediately after the corresponding return from $m$ as well”

$\omega$-VPA:
“If $p$ holds true immediately after entering module $m$, it shall hold immediately after the corresponding return from $m$ as well”

**VLTL:**

$$(\alpha; \text{true})|\alpha\rangle\text{false}$$

with visibly rational expression $\alpha$ below:

$$[(p \cup q)^* \text{call}_m [(q \Box) \cup (p \Box p)] \text{return}_m (p \cup q)^*] \Box \lor \Box (p \cup q)^*$$
## Our Results

<table>
<thead>
<tr>
<th></th>
<th>validity</th>
<th>model-checking</th>
<th>infinite games</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTL</td>
<td>PSPACE</td>
<td>PSPACE</td>
<td>2\text{ExpTime}</td>
</tr>
<tr>
<td>LDL</td>
<td>PSPACE</td>
<td>PSPACE</td>
<td>2\text{ExpTime}</td>
</tr>
<tr>
<td>Logic</td>
<td>validity</td>
<td>model-checking</td>
<td>infinite games</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>LTL</td>
<td>$\text{PSpace}$</td>
<td>$\text{PSpace}$</td>
<td>$2\text{ExpTime}$</td>
</tr>
<tr>
<td>LDL</td>
<td>$\text{PSpace}$</td>
<td>$\text{PSpace}$</td>
<td>$2\text{ExpTime}$</td>
</tr>
<tr>
<td>VLDL</td>
<td>$\text{ExpTime}$</td>
<td>$\text{ExpTime}$</td>
<td>$3\text{ExpTime}$</td>
</tr>
<tr>
<td>VLTL</td>
<td>$\text{ExpTime}$</td>
<td>$\text{ExpTime}$</td>
<td>$?$</td>
</tr>
</tbody>
</table>
## Our Results

<table>
<thead>
<tr>
<th>Logic</th>
<th>validity</th>
<th>model-checking</th>
<th>infinite games</th>
</tr>
</thead>
<tbody>
<tr>
<td>LTL</td>
<td>PSPACE</td>
<td>PSPACE</td>
<td>2ExpTime</td>
</tr>
<tr>
<td>LDL</td>
<td>PSPACE</td>
<td>PSPACE</td>
<td>2ExpTime</td>
</tr>
<tr>
<td>VLDL</td>
<td>ExpTime</td>
<td>ExpTime</td>
<td>3ExpTime</td>
</tr>
<tr>
<td>VLTL</td>
<td>ExpTime</td>
<td>ExpTime</td>
<td>?</td>
</tr>
<tr>
<td>VLDL\text{exp}</td>
<td>ExpTime</td>
<td>ExpTime</td>
<td>3ExpTime</td>
</tr>
</tbody>
</table>