Reducing CTL-Live Model Checking to First-Order Logic Validity Checking

Amirhossein Vakili and Nancy A. Day

Cheriton School of Computer Science

24 October 2014
Model Checking based on SAT/SMT Solving

- Focus on safety properties
- Iteratively calls the solver
Our Result: CTL-Live Model Checking as FOL Validity

Liveness Property: Is X always reachable?

Focus on liveness properties
Solved by first-order logic deduction techniques (e.g., SMT solvers)
No need for abstraction or invariant generation
CTL-Live includes CTL connectives that are defined using *the least fixpoint operator* of mu-calculus.

<table>
<thead>
<tr>
<th>Temporal part</th>
<th>Propositional part</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \varphi ) ::= ( \pi \mid \varphi_1 \lor \varphi_2 \mid \varphi_1 \land \varphi_2 )</td>
<td>( \pi ) ::= ( P \mid \neg \pi \mid \pi_1 \lor \pi_2 )</td>
</tr>
<tr>
<td>( ::= )</td>
<td>( )</td>
</tr>
<tr>
<td>( ::= )</td>
<td>( )</td>
</tr>
<tr>
<td>( )</td>
<td>( )</td>
</tr>
<tr>
<td>( )</td>
<td>( )</td>
</tr>
<tr>
<td>( )</td>
<td>( )</td>
</tr>
<tr>
<td>( )</td>
<td>( )</td>
</tr>
<tr>
<td>( )</td>
<td>( )</td>
</tr>
<tr>
<td>( )</td>
<td>( )</td>
</tr>
<tr>
<td>where ( P ) is a labelling predicate.</td>
<td>where ( P ) is a labelling predicate.</td>
</tr>
</tbody>
</table>
CTL-Live includes CTL connectives that are defined using the least fixpoint operator of mu-calculus.

<table>
<thead>
<tr>
<th>Temporal part</th>
<th>Propositional part</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varphi ::= \pi \mid \varphi_1 \lor \varphi_2 \mid \varphi_1 \land \varphi_2$</td>
<td>$\pi ::= P \mid \neg \pi \mid \pi_1 \lor \pi_2$</td>
</tr>
<tr>
<td>$::= \text{EX} \varphi \mid \text{AX} \varphi \mid \text{EF} \varphi \mid \text{AF} \varphi$</td>
<td></td>
</tr>
<tr>
<td>$::= \varphi_1 \text{EU} \varphi_2 \mid \varphi_1 \text{AU} \varphi_2$</td>
<td></td>
</tr>
</tbody>
</table>

where $P$ is a labelling predicate.

In CTL-Live

- $\text{AF} \ P$
- $(\text{EF} \neg P) \ \text{AU} \ (\text{AX} \ Q)$
CTL-Live includes CTL connectives that are defined using the least fixpoint operator of mu-calculus.

<table>
<thead>
<tr>
<th>Temporal part</th>
<th>Propositional part</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \varphi ::= \pi \mid \varphi_1 \lor \varphi_2 \mid \varphi_1 \land \varphi_2 )</td>
<td>( \pi ::= P \mid \neg \pi \mid \pi_1 \lor \pi_2 )</td>
</tr>
<tr>
<td>( ::= )</td>
<td>where ( P ) is a labelling predicate.</td>
</tr>
<tr>
<td>( ::= )</td>
<td>( )</td>
</tr>
<tr>
<td>( )</td>
<td>( )</td>
</tr>
</tbody>
</table>

**In CTL-Live**

- \( \text{AF} \ P \)
- \( (\text{EF}\neg P) \ \text{AU} \ (\text{AX} \ Q) \)

**Not In CTL-Live**

- \( \neg(\text{AF} \ P) \)
- \( \text{AG} \ P \)
Symbolic Kripke Structures in FOL

initial $\rightarrow c = 0 \rightarrow c = 3 \rightarrow c = 4 \rightarrow \cdots$

$\rightarrow c = 2 \rightarrow c = 5 \rightarrow \cdots$

$\rightarrow c = 6 \rightarrow \cdots$

$S = \{0, 1, 2, 3, \ldots\}$ state space

$S^0(c) \iff c = 0$ initial states

$N(c, c') \iff c' = c + 2 \lor c' = c + 3$ next-state relation

Notation

$\text{symbolic}(K) \models AF_{c > 3}$

$AF_{c > 3} = \{0, 1, 2, \ldots\}$
Symbolic Kripke Structures in FOL

- $S = \{0, 1, 2, 3, ..\}$
- $S_0(c) \iff c = 0$
- $N(c, c') \iff c' = c + 2 \lor c' = c + 3$
Symbolic Kripke Structures in FOL

\[ S = \{0, 1, 2, 3, \ldots\} \]

\[ S_0(c) \iff c = 0 \]

\[ N(c, c') \iff c' = c + 2 \lor c' = c + 3 \]

**Notation**

- \( \text{symbolic}(K) \models c \ AF \ c > 3 \)
- \([AF \ c > 3] = \{0, 1, 2, \ldots\}\)
Intuition: States Satisfying $\text{AF } P$

According to encoding of $\text{AF}$ in mu-calculus, $[\text{AF } P]$ is the smallest set $Y$ that satisfies:

(1) $\forall s \cdot P(s) \Rightarrow Y(s)$

(2) $\forall s \cdot (\forall s' \cdot N(s, s') \Rightarrow Y(s')) \Rightarrow Y(s)$
Intuition: States Satisfying $\textbf{AF} \ P$

According to encoding of $\textbf{AF}$ in mu-calculus, $[\textbf{AF} \ P]$ is the \textbf{smallest} set $Y$ that satisfies:

1. $\forall s \bullet P(s) \Rightarrow Y(s)$  
2. $\forall s \bullet (\forall s' \bullet N(s, s') \Rightarrow Y(s')) \Rightarrow Y(s)$

State Space
Intuition: States Satisfying $\text{AF} \ P$

According to encoding of $\text{AF}$ in mu-calculus, $[\text{AF} \ P]$ is the smallest set $Y$ that satisfies:

1. $\forall s \cdot P(s) \Rightarrow Y(s)$
2. $\forall s \cdot (\forall s' \cdot N(s, s') \Rightarrow Y(s')) \Rightarrow Y(s)$

State Space

$Y_1$
Intuition: States Satisfying \( \text{AF} P \)

According to encoding of \( \text{AF} \) in mu-calculus, \([\text{AF} P]\) is the \textbf{smallest} set \( Y \) that satisfies:

\[
(1) \forall s \cdot P(s) \implies Y(s) \\
(2) \forall s \cdot (\forall s' \cdot N(s, s') \implies Y(s')) \implies Y(s)
\]
Intuition: States Satisfying $\textbf{AF} P$

According to encoding of $\textbf{AF}$ in mu-calculus, $[\textbf{AF} P]$ is the smallest set $Y$ that satisfies:

(1) $\forall s \bullet P(s) \Rightarrow Y(s)$

(2) $\forall s \bullet (\forall s' \bullet N(s, s') \Rightarrow Y(s')) \Rightarrow Y(s)$

State Space

$Y_1 \quad Y_2 \quad Y_3$
Intuition: States Satisfying $\text{AF } P$

According to encoding of $\text{AF}$ in mu-calculus, $[\text{AF } P]$ is the smallest set $Y$ that satisfies:

1. $\forall s \cdot P(s) \Rightarrow Y(s)$
2. $\forall s \cdot (\forall s' \cdot N(s, s') \Rightarrow Y(s')) \Rightarrow Y(s)$
Intuition: States Satisfying $\operatorname{AF} P$

According to encoding of $\operatorname{AF}$ in mu-calculus, $[\operatorname{AF} P]$ is the smallest set $Y$ that satisfies:

1. $\forall s \cdot P(s) \Rightarrow Y(s)$
2. $\forall s \cdot (\forall s' \cdot N(s, s') \Rightarrow Y(s')) \Rightarrow Y(s)$

$$[\operatorname{AF} P] = \bigcap_{Y \in \Theta} Y$$

where $\Theta = \{Y \text{ satsifying (1), (2)}\}$
Intuition: Model Checking \( AF \ P \)

Model checking is about a subset relation, \( S_0 \subseteq \{AF \ P\} \):

\[
S_0 \subseteq \bigcap_{Y \in \Theta} Y
\]
Intuition: Model Checking \( \textbf{AF} \ P \)

Model checking is about a subset relation, \( S_0 \subseteq [\textbf{AF} \ P] \):

\[
S_0 \subseteq \bigcap_{Y \in \Theta} Y \quad \text{iff} \quad \forall Y \in \Theta \bullet S_0 \subseteq Y
\]
Intuition: Model Checking $\textbf{AF} \ P$

Model checking is about a subset relation, $S_0 \subseteq [\textbf{AF} \ P]$:

$$S_0 \subseteq \bigcap_{Y \in \Theta} Y \iff \forall Y \in \Theta \bullet S_0 \subseteq Y$$

- Higher-order universal quantifier
Model checking is about a subset relation, $S_0 \subseteq [\text{AF } P]$: 

\[ S_0 \subseteq \bigcap_{Y \in \Theta} Y \quad \text{iff} \quad \forall Y \in \Theta \bullet S_0 \subseteq Y \]

- Higher-order universal quantifier
- First-order logic formula
Intuition: Model Checking $\mathbf{AF} \, P$

Model checking is about a subset relation, $S_0 \subseteq [\mathbf{AF} \, P]$:

$$S_0 \subseteq \bigcap_{Y \in \Theta} Y \iff \forall Y \in \Theta \bullet S_0 \subseteq Y$$

- Higher-order universal quantifier
- First-order logic formula

**Definition (FOL Validity)**

$\Gamma \models \Phi$ iff every interpretation that satisfies $\Gamma$ also satisfies $\Phi$. 
Intuition: Model Checking $\text{AF } P$

Model checking is about a subset relation, $S_0 \subseteq [\text{AF } P]$:

$$S_0 \subseteq \bigcap_{Y \in \Theta} Y \iff \forall Y \in \Theta \bullet S_0 \subseteq Y$$

- Higher-order universal quantifier
- First-order logic formula

Definition (FOL Validity)

$$\Gamma \models \Phi \text{ iff every interpretation that satisfies } \Gamma \text{ also satisfies } \Phi.$$
Our Result

**Reduction Procedure:**

**INPUT:**

- $\text{symbolic}(K)$: symbolic representation of a Kripke structure.
- $\varphi$: a CTL-Live formula.

**OUTPUT:**

$$\text{symbolic}(K) \cup \text{CTLL2FOL}(\varphi) \models S_0 \subseteq \lceil \varphi \rceil$$

**Theorem (Reduction of CTL-Live Model Checking to FOL Validity)**

$$\text{symbolic}(K) \models_c \varphi$$

iff

$$\text{symbolic}(K) \cup \text{CTLL2FOL}(\varphi) \models S_0 \subseteq \lceil \varphi \rceil$$
Our Result

**Reduction Procedure:**
**INPUT:**
- \( \text{symbolic}(K) \): symbolic representation of a Kripke structure.
- \( \varphi \): a CTL-Live formula.

**OUTPUT:**
- \( \text{symbolic}(K) \cup \text{CTLL2FOL}(\varphi) \models S_0 \subseteq [\varphi] \)

**Example:**

\[
\forall c \cdot S_0(c) \iff c = 0 \\
\forall c, c' \cdot N(c, c') \iff c' = c + 2 \lor c' = c + 3 \\
\forall c \cdot c > 3 \Rightarrow Y(c) \\
\forall c \cdot (\forall c' \cdot N(c, c') \Rightarrow Y(c')) \Rightarrow Y(c) \models S_0 \subseteq Y
\]
Our Result

**Reduction Procedure:**

**INPUT:**
- \(symbolic(K)\): symbolic representation of a Kripke structure.
- \(\varphi\): a CTL-Live formula.

**OUTPUT:**
- \(symbolic(K) \cup \text{CTLL2FOL}(\varphi) \models S_0 \subseteq \llbracket \varphi \rrbracket\)

**Example:**

\[\forall c \bullet S_0(c) \Leftrightarrow c = 0\]
\[\forall c, c' \bullet N(c, c') \Leftrightarrow c' = c + 2 \lor c' = c + 3\]
\[\forall c \bullet c > 3 \Rightarrow Y(c)\]
\[\forall c \bullet (\forall c' \bullet N(c, c') \Rightarrow Y(c')) \Rightarrow Y(c) \models S_0 \subseteq Y\]
Our Result

**Reduction Procedure:**

**INPUT:**
- $\textit{symbolic}(K)$: symbolic representation of a Kripke structure.
- $\varphi$: a CTL-Live formula.

**OUTPUT:**
- $\textit{symbolic}(K) \cup \text{CTLL2FOL}(\varphi) \models S_0 \subseteq [\varphi]$

**Example:**

\[
\forall c \cdot S_0(c) \iff c = 0
\]
\[
\forall c, c' \cdot N(c, c') \iff c' = c + 2 \lor c' = c + 3
\]
\[
\forall c \cdot c > 3 \Rightarrow Y(c)
\]
\[
\forall c \cdot (\forall c' \cdot N(c, c') \Rightarrow Y(c')) \Rightarrow Y(c) \models S_0 \subseteq Y
\]
Based on this result, we used Z3 and CVC4 to model check CTL-Live properties of 4 infinite systems.

Case studies were from different domains.

SMT solvers are efficient in model checking CTL-Live properties.

Conclusion

- Presented CTL-Live, a fragment of CTL such that its model checking is reducible to FOL validity.
  - No need for abstraction or invariant generation
  - Use state-of-the-art FOL reasoners for model checking
  - Only FOL reasoning is required for verification