Theoretical results around Electrum

Julien Brunel    David Chemouil    Denis Kuperberg

ONERA/DTIM - IRIT

Séminaire DTIM
11/05/2015
Toulouse
Introduction

Alloy Language
- Specification language based on First-Order Logic
- Inspired by UML, user-friendly
- Arbitrary predicates → Expressivity

Alloy Analyzer
- Bounded verification → Decidability
- Use of SAT solvers → Efficiency, quick feedback
Example of Alloy Specification:

```alloy
open util/ordering [Book] as BookOrder

sig Addr {}
sig Name {}
sig Book {
    names: set Name,
    addr: names→some Addr
}

pred add [b1, b2: Book, n: Name, a: Addr] {
    b2.addr = b1.addr + n→a
}

pred del [b1, b2: Book, n: Name, a: Addr] {
    b2.addr = b1.addr − n→a
}

fact traces {
    all b: Book−last | let bnext = b.BookOrder/next | some n: Name, a: Addr |
    add [b, bnext, n, a] or del [b, bnext, n, a]
}
```

One object book for each time instant. Tedious way of modeling time and reasoning about it.
Model finder

// Show a model where some name has two different addresses
run {some b: Book, n: Name, disj a1, a2: Addr
    | a1 in n.(b.addr) and a2 in n.(b.addr)}

Property checker

assert delUndoesAdd {
    all b1, b2, b3: Book, n: Name, a: Addr |
    no n.(b1.addr) and add [b1, b2, n, a] and del [b2, b3, n, a]
    implies b1.addr = b3.addr
}
check delUndoesAdd
Electrum: Alloy + new dedicated time operators like \( \text{′} \) (value at the next instant) and always:

\[
\text{sig} \quad \text{Addr} \{\}
\]

\[
\text{sig} \quad \text{Name} \{
\quad \text{var} \ \text{addr} : \text{set Addr}
\}
\]

\[
\text{pred} \quad \text{add} \ [n: \text{Name}, a: \text{Addr}] \{ \quad \text{addr′ = addr +n→a}\}
\]

\[
\text{pred} \quad \text{del} \ [n: \text{Name}, a: \text{Addr}] \{ \quad \text{addr′ = addr - n→a}\}
\]

\[
\text{fact} \quad \text{traces} \{ \quad \text{always} \quad \{ \quad \text{some} \ n: \text{Name}, a: \text{Addr} \ | \ \text{add} \ [n, a] \ \text{or} \ \text{del} \ [n, a]\} \}
\]

Infinite number of time instants, that can be referred to easily with a specialized syntax.
**Abstraction:** The logic FO-LTL.

**LTL:** Good properties of expressivity and complexity, widely used in verification to model infinite time traces.

The logic **FO-LTL**:

\[ \varphi ::= (x_1 = x_2) \mid P_i(x_1, \ldots, x_n) \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists x \varphi \mid \text{next}\varphi \mid \varphi \text{until}\varphi. \]

We also define \( \text{eventually}\varphi = \text{trueuntil}\varphi \) and \( \text{always}\varphi = \neg \text{eventually}(\neg \varphi) \).

We use FO-LTL as underlying logic of the new language **Electrum**.

- First-Order variables \( x_i \): finite domain
- Implicit time: infinite domain \( \mathbb{N} \)

What is the theoretical cost of adding LTL?
NSAT Problem: Given $\varphi$ and $N$, is there a model for $\varphi$ of First-Order domain of size at most $N$?

Parameters:

- **Logic:** FO versus FO-LTL
- **Encoding of $N$:** unary versus binary
- **Rank of formulas (nested quantifiers):** bounded ($\bot$) versus unbounded ($\top$).
NSAT Problem: Given $\varphi$ and $N$, is there a model for $\varphi$ of First-Order domain of size at most $N$?

Parameters:
- Logic: $\text{FO}$ versus $\text{FO-LTL}$
- Encoding of $N$: unary versus binary
- Rank of formulas (nested quantifiers): bounded ($\bot$) versus unbounded ($\top$).

Theorem

<table>
<thead>
<tr>
<th></th>
<th>$N$ unary</th>
<th>$N$ binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{FO} \bot$</td>
<td>$\text{NP-complete}$</td>
<td>$\text{NEXPTIME-complete}$</td>
</tr>
<tr>
<td>$\text{FO} \top$</td>
<td>$\text{NEXPTIME-complete}$</td>
<td>$\text{NEXPTIME-complete}$</td>
</tr>
<tr>
<td>$\text{FO-LTL} \bot$</td>
<td>$\text{PSPACE-complete}$</td>
<td>$\text{EXPSPACE-complete}$</td>
</tr>
<tr>
<td>$\text{FO-LTL} \top$</td>
<td>$\text{EXPSPACE-complete}$</td>
<td>$\text{EXPSPACE-complete}$</td>
</tr>
</tbody>
</table>
Algorithms for membership

**FO cases:** we use a naive non-deterministic algorithm that

- guesses a structure, i.e. writes the value of predicates for each possible input,
- verifies the formula on it.

**FO-LTL cases:**

- Use naked structure $S = \{1, \ldots, N\}$
- Expand $\varphi$ into a LTL formula $\psi$, by turning FO quantifiers into disjunctions/conjunctions over $S$.
- Alphabet of $\psi$ is
  \[ A = \{ P(s_1, \ldots, s_k) \mid P \text{ predicate of } \varphi, s_i \in S \} \]
- Check that $S \models \psi$: this is PSPACE in $|S| + |\psi|$. 
Proof scheme for hardness

**Idea**: encode runs of Turing Machines via formulas.

For FO, unbounded rank, binary encoding:

**Reduction**:

- Start from non-deterministic $M$ running in time $2^n$ on inputs of size $n$. States $Q$ and alphabet $A$.
- Consider the first-order structure $\{1, \ldots, 2^n\}$ with predicate successor, representing both time and space of the machine.
- Predicate $a(x, t)$ with $a \in A$: the cell $x$ is labeled $a$ at time $t$
- Predicate $q(x, t)$: $M$ is in state $q$ in position $x$ at time $t$
For any word $u$ of size $n$, we can now write a formula $\varphi_u$ of size polynomial in $n$, stating that:

- The initial configuration of the tape is $u$:
  $$a_1(1, 1) \land a_2(2, 1) \land \cdots \land a_n(n, 1)$$

- For all time $t$, the tape is updated from $t$ to $t + 1$ according to the transition table of $M$

- there is a time $t_f$ where $M$ is in its accepting state.

**Correctness:** $\varphi_u$ has a model of size $2^n \iff u$ is accepted by $M$

Size $2^n$ is given in binary $\rightarrow$ polynomial reduction.
For any word $u$ of size $n$, we can now write a formula $\varphi_u$ of size polynomial in $n$, stating that:

- The initial configuration of the tape is $u$:
  \[ a_1(1, 1) \land a_2(2, 1) \land \cdots \land a_n(n, 1) \]
- For all time $t$, the tape is updated from $t$ to $t + 1$ according to the transition table of $M$
- there is a time $t_f$ where $M$ is in its accepting state.

**Correctness:** $\varphi_u$ has a model of size $2^n \iff u$ is accepted by $M$

Size $2^n$ is given in binary $\rightarrow$ polynomial reduction.

**Extension to FO-LTL:** LTL uses implicit time $\rightarrow$ we can start from an EXPSPACE machine.
Constraint on transitions is now of the form
\[
\text{always}(\forall x, q(x) \implies \text{next}\varphi_q(x))
\]
Tricky case: unbounded rank but unary $N$.
$\rightarrow$ We can no longer use the domain as a model for the tape.
**Tricky case:** unbounded rank but unary $N$.

$\rightarrow$ We can no longer use the domain as a model for the tape.

**Solution:** Use a structure of size 2, and binary encoding to point to a cell or time instant: $a(\vec{x}, \vec{t})$ for FO and $a(\vec{x}')$ for FO-LTL.

**Example:** For size 8, $a(0, 1, 1, 1, 0, 1)$ means that the 3$^{th}$ cell is labeled by $a$ at instant 5.
Finite Model Theory

**Finite Model Property:** If there is a model there is a finite one.

**FO Fragments with FMP:**

- $[\exists^* \forall^*, \text{all}] \equiv$ (Ramsey 1930)
- $[\exists^* \forall^* \exists^*, \text{all}] \equiv$ (Ackermann 1928)
- $[\exists^*, \text{all}, \text{all}] \equiv$ (Gurevich 1976)
- $[\exists^* \forall, \text{all}, (1)] \equiv$ (Grädel 1996)
- $\mathit{FO}_2$ (Mortimer 1975): 2 variables.

**Theorem**

*Adding next, eventually preserves FMP if the fragment imposes no constraint on the number and arity of predicates/functions.*

**True** for all above fragments except Grädel: only one function of arity one.
Axioms of infinity

In general, adding LTL allows to write axioms of infinity:

With one existential variable:

\[
\text{always}(\exists x. P(x) \land \text{next}(\text{always}\neg P(x))).
\]

Without nesting quantifiers in temporal operators:

\[
\forall x \exists y. P(c) \land \text{always}(P(x) \Rightarrow \text{next}(P(y) \land \text{always}\neg P(x))).
\]

Without \text{always}:

\[
\forall x \exists y. P(c) \land ((P(x) \land P(y))\text{until}(\neg P(x) \land P(y))).
\]
Conclusion

Theoretical study of FO-LTL versus FO
  ▶ Complexity
  ▶ Finite model property

On-going work with Univ. of Minho/IRIT
  ▶ Implementation of different verification procedures for Electrum:
    • Reduce to LTL satisfiability
    • Reduce to Alloy
  ▶ Use of efficient solvers
  ▶ Comparison with TLA and B