Model Checking with Computation Tree Logic

Joe Scott

February 17, 2012
Different Approaches to Verification

- **Proof-Based**
  - Representation:
    - System description is a set of formulas $\Gamma$ in a suitable logic.
    - Specification is another formula $\phi$
  - Find a proof that $\Gamma \vdash \phi$
    - Deductive
    - Usually requires guidance from the user

- **Model-Based**
  - Representation:
    - System description is a model $\mathcal{M}$ of a suitable logic.
    - Specification still a formula $\phi$
  - Determine whether $\mathcal{M} \models \phi$
    - Algorithmic
    - Automatic
Why model checking?

- Given a logical proof system that is sound and complete\(^1\): \(\Gamma \vdash \phi\) (provability) holds iff \(\Gamma \models \phi\) (semantic entailment).
- Semantic entailment means for all models \(\mathcal{M}\):
  if for all \(\psi \in \Gamma\) we have \(\mathcal{M} \models \psi\), then \(\mathcal{M} \models \phi\).

**Intuition**
A verification method based on a single model \(\mathcal{M}\) should be simpler than a method based on a potentially infinite class of them.

---
\(^1\)Of course, Hoare Logic is not complete.
Temporal Logic

- Classical propositional and predicate logics are static
  - formulas are always true or false
- In modal logic, truth is dynamic
  - models contain several states
  - a formula may be true in some states, false in others
- Temporal logic is a modal logic with a semantics based on “when”
  - a path is a sequence of time instances (states)

<table>
<thead>
<tr>
<th>Temporal model</th>
<th>Linear</th>
<th>Branching</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path Quantification</td>
<td>set of paths</td>
<td>tree</td>
</tr>
<tr>
<td>∀ (implicit)</td>
<td>LTL</td>
<td>∀ or ∃ (explicit)</td>
</tr>
<tr>
<td>LTL</td>
<td>CTL</td>
<td>CTL*</td>
</tr>
</tbody>
</table>
Model: Transition System

\[ M = \langle S, \rightarrow, L \rangle \]

- (finite) set of states \( S \)
- transition relation \( \rightarrow \):
  - Binary relation on \( S \)
  - Every \( s \in S \) has some \( s' \in S \) such that \( s \rightarrow s' \)
- labelling function \( L : S \rightarrow \mathcal{P}(\text{atoms}) \)
CTL Syntax

Valid formulas:
- True, False
- Any atomic proposition $p$
- For valid subformulas $\phi_1, \phi_2$:
  $$ \neg \phi_1, \phi_1 \land \phi_2, \phi_1 \lor \phi_2, \phi_1 \Rightarrow \phi_2, \ldots $$
- temporal formula:

  ![Diagram of CTL syntax]

  **path quantifier:**
  - All paths
  - Exists a path

  **temporal operator:**
  - next state
  - some future state
  - all future states (Globally)
  - all states Until (binary)

  **subformula**
CTL Equivalences

- $\neg AF\phi \equiv EG\neg\phi$
- $\neg EF\phi \equiv AG\neg\phi$
- $\neg AX\phi \equiv EX\neg\phi$
- $AF\phi \equiv A[\top U\phi]$
- $EF\phi \equiv E[\top U\phi]$

Theorem (Adequate sets of CTL connectives\textsuperscript{a})


A set of temporal connectives in CTL is adequate iff it contains:

- at least one of \{AX, EX\}
- at least one of \{EG, AF, AU\}
- EU
Labelling Algorithm for \( \{ AF, EU, EX \} \)

Starting from innermost subformulas:
- \( \bot \): do nothing
- \( p \): label \( s \) if \( p \in L(s) \)
- \( \phi_1 \land \phi_2 \): label \( s \) if \( s \) is already labelled with \( \phi_1 \) and \( \phi_2 \)
- \( \neg \phi_1 \): label all \( s \) not already labelled \( \phi_1 \)
- \( AF\phi_1 \):
  1. If any \( s \) is labelled \( \phi_1 \), label it \( AF\phi_1 \)
  2. Label any state \( AF\phi_1 \) if all its successor states are labelled \( AF\phi_1 \)
  3. Repeat 2 until no change
- \( E[\phi_1 U \phi_2] \):
  1. If any \( s \) is labelled \( \phi_2 \), label it \( E[\phi_1 U \phi_2] \)
  2. For any \( s \) labelled \( \phi_1 \), label \( s \) if it has a successor labelled \( E[\phi_1 U \phi_2] \)
  3. Repeat 2 until no change
- \( EX\phi_1 \): label any state that has a successor labelled \( \phi_1 \)
A final example

Does $AF \ AG p$ hold? No.

LTL formula $F \ G p$ does hold.
References

- Today’s paper:
  Clarke, Emerson, Sistla
  Automatic verification of finite-state concurrent systems using temporal logic specifications.

- Supplementary slides courtesy:

- Other material based on: