How is Model-Checking Done?

- Semantics of propositions
- Tableau-based checking:
  - Notion of tableau
  - Rules
  - Example
- Implementation of model-checking
- Running times

Terminology

$\mathcal{A}$ - a set of *atomic formulas A* ...
$\mathcal{V}$ (disjoint from $\mathcal{A}$) - a set of *propositional variables X* ...
Act - a set of *actions a* ...
Formulas are Φ ...
$\mathcal{T}$ - a set of states

- $\Gamma \prec \Phi$ if Γ is a strict subformula of Φ.

environment - a mapping of variables to sets of states as a means of interpreting free propositional variables.

- $e[X \mapsto S]$ - the environment e with X "updated" to S.
- Use *sequents* of the form $H \vdash s \in \Phi$, where s is a state, Φ is a formula, and H is a set of *hypotheses* of the form $s\prime : \Gamma$, for $s\prime$ a state and Γ a *closed recursive formula*.

Semantics of propositions

$$
\begin{array}{rcl}
[\![A]\!]e & = & V(A) \\
[\![X]\!]e & = & e(X) \\
[\![\neg\Phi]\!]e & = & \mathcal{T} - [\![\Phi]\!]e \\
[\![\Phi_1 \vee \Phi_2]\!]e & = & [\![\Phi_1]\!]e \cup [\![\Phi_2]\!]e \\
[\![< a > \Phi]\!]e & = & \pi_a([\![\Phi]\!]e), \text{ where} \\
 & & \pi_a(S) = \{s\prime \mid \exists s \in S. s\prime \xrightarrow{a} s\} \\
[\![\nu X.\Phi]\!]e & = & \bigcup\{S \subseteq \mathcal{T} \mid S \subseteq [\![\Phi]\!]e[X \mapsto S]\}
\end{array}
$$

Idea of tableau

- Theorem: $H \vdash s \in \Phi$ has a successful tableau if and only if $H \vdash s \in \neg\Phi$ has no successful tableau.
- Idea: start with property (or negated property), apply rules R1-R8 and DR1-DR3 (below) in top-down fashion until *all* leaves are successful. A leaf is successful if and only if one of the following holds:

1. $\Phi \in \mathcal{A}$ and $s \in (V(\Phi))$.
2. Φ is $\neg A$ for some $A \in \mathcal{A}$ and $s \notin V(A)$.
3. Φ is $\neg < a > \Phi\prime$ for some a and $\Phi\prime$.
4. Φ is $\nu X.\Phi\prime$ for some X and $\Phi\prime$.
5. Sequents of form $H \vdash s \in Tr^{ue}$ are successful.
6. Leaves of the form $H \vdash s \in [a]\Phi$ are successful.

Note: $H \vdash s \in \neg < a > \Phi$ is a leaf only when s has no a-derivatives, while $H \vdash s \in \nu X.\Phi$ is a leaf only when $s : \nu X.\Phi \in H$.

Rules

see Figure 3 on p. 730 of Acta Informatica paper

Rules (Cont'd)

see Figure 4 on p. 732 of Acta Informatica paper

Rules, Etc.

R7 and R8 require that in order to establish that a state enjoys a (negated) recursive property, it is sufficient to establish that it enjoys the (negated) unrolling of the property, provided that the assumptions involving the formulas having the recursive formula as a subformula are removed or *discharged* from the hypothesis list.

Other results:
1. (Finiteness) If models are finite, their tableaux are finite.
2. (Soundness and completeness) $H \vdash s \in \Phi$ has a successful tableau if and only if $s \in [\![\Phi]\!]^H$.

Example

See Figure 5 on p. 732 of Acta Informatica paper

Simple Implementation of model-checking

fun *check1′*($H \vdash s \in \Phi$) =
  case Φ is
  $A \in \mathcal{A} \rightarrow$ return ($s \in \mathcal{V}(\mathcal{A})$)
  $X \in \mathcal{V} \rightarrow$ error
  $\neg\Phi' \rightarrow$ return not (*check1′*($H \vdash s \in \Phi'$))
  $\Phi_1 \vee \Phi_2 \rightarrow$ return (*check1′*($H \vdash s \in \Phi_1$)
        or *check1′*($H \vdash s \in \Phi_2$))
  $< a > \Phi' \rightarrow$ for each $s' \in \{s' s \xrightarrow{s'} s'\}$ do
        if *check1′*($H \vdash s' \in \Phi'$) then
            return true;
        else return false
  $\nu X.\Phi' \rightarrow$ let $H' = \{s' : \Gamma \mid \Phi \neg \prec \Gamma\}$ in
        return (*check1′*($H \cup \{s : \Phi\} \vdash$
            $s \in \Phi'[\Phi/X]$))
end
fun *check1*($s \in \Phi$) = *check1′*($\emptyset \vdash s \in \Phi$)

Running times

• Algorithm has exponential running time even for formulas having no recursive subformulas, owing to the possibility of nested modal operators.

• Possible optimization: store results of sequents whose truth has already been determined

• Running time is $O((|S| \times |\Phi|)^{id(\Phi)+1})$:
  – $id(\Phi)$ – *interconnection depth* of Φ, measure of the degree of mutual recursion in Φ
  – Φ – formula under verification
  – S – number of states in transition system