CS477 Formal Software Development Methods

Elsa L Gunter
2112 SC, UIUC
egunter@illinois.edu
http://courses.engr.illinois.edu/cs477

Slides based in part on previous lectures by Mahesh Vishwanathan, and by Gul Agha

April 16, 2014

Formal LTL Semantics

- $\sigma \models p$ iff $q_0 \models p$
- $\sigma \models \neg\varphi$ iff $\sigma \not\models \varphi$
- $\sigma \models \varphi \wedge \psi$ iff $\sigma \models \varphi$ and $\sigma \models \psi$.
- $\sigma \models \varphi \vee \psi$ iff $\sigma \models \varphi$ or $\sigma \models \psi$.
- $\sigma \models \circ\varphi$ iff $\sigma^1 \models \varphi$
- $\sigma \models \varphi\mathcal{U}\psi$ iff for some k, $\sigma^k \models \psi$ and for all $i < k$, $\sigma^i \models \varphi$
- $\sigma \models \varphi\mathcal{V}\psi$ iff for some k, $\sigma^k \models \varphi$ and for all $i \leq k$, $\sigma^i \models \psi$,
  or for all i, $\sigma^i \models \psi$.
- $\sigma \models \Box\varphi$ if for all i, $\sigma^i \models \psi$
- $\sigma \models \Diamond\varphi$ if for some i, $\sigma^i \models \psi$

Some More Eqivalences

- $\Box\varphi \Leftrightarrow \varphi \wedge \circ\Box\varphi$
- $\Diamond\varphi \Leftrightarrow \varphi \vee \circ\Diamond\varphi$
- $\varphi\,\mathcal{U}\,\psi \Leftrightarrow \phi \vee (\psi \wedge \circ(\varphi\,\mathcal{U}\,\psi))$
- $\varphi\,\mathcal{V}\,\psi \Leftrightarrow (\varphi \wedge \psi) \vee (\varphi \wedge \circ(\varphi\,\mathcal{V}\,\psi)$
- $\Box$, $\Diamond$, $\mathcal{U}$, $\mathcal{V}$ may all be understood recursively, by what they state about right now, and what they state about the future
- Caution: $\Box$ vs $\Diamond$, $\mathcal{U}$ vs $\mathcal{V}$ differ in there limit behavior

Traffic Light Example

Basic Behavior:

- $\Box((NSC = Red) \vee (NSC = Green) \vee (NSC = Yellow))$
- $\Box((NSC = Red) \Rightarrow ((NSC \neq Green) \wedge (NSC \neq Yellow))$
- Similarly for *Green* and *Red*
- $\Box(((NCS = Red) \wedge \circ(NCS \neq Red)) \Rightarrow \circ(NCS = Green))$
- Same as $\Box((NCS = Red) \Rightarrow ((NCS = Red)\,\mathcal{U}\,(NCS = Green)))$
- $\Box(((NCS = Green) \wedge \circ(NCS \neq Green)) \Rightarrow \circ(NCS = Yellow))$
- $\Box(((NCS = Yellow) \wedge \circ(NCS \neq Yellow)) \Rightarrow \circ(NCS = Red))$
- Same for *EWC*

Traffic Light Example

Basic Safety

- $\Box((NSC = Red) \vee (EWC = Red))$
- $\Box(\ ((NSC = Red) \wedge (EWC = Red))\ \mathcal{V}$
  $((NSC \neq Green) \Rightarrow (\circ(NSC \neq Green))))$

Basic Liveness

- $(\Diamond(NSC = Red)) \wedge (\Diamond(NSC = Green)) \wedge (\Diamond(NSC = Yellow))$
- $(\Diamond(EWC = Red)) \wedge (\Diamond(EWC = Green)) \wedge (\Diamond(EWC = Yellow))$

Proof System for LTL

- First Step: View $\varphi \mathcal{V} \psi$ as macro: $\varphi \mathcal{V} \psi = \neg((\neg\varphi) \mathcal{U} (\neg\psi))$
- Second Step: Extend all rules of Prop Logic to LTL
- Third Step: Add one more rule: $\dfrac{\varphi}{\Box\varphi}$ Gen
- Fourth Step: Add a collection of axioms (a sufficient set of 8 exists)
  - A1: $\Box\varphi \Leftrightarrow \neg(\Diamond(\neg\varphi))$
  - A2: $\Box(\varphi \Rightarrow \psi) \Rightarrow (\Box\varphi \Rightarrow \Box\psi)$
  - A3: $\Box\varphi \Rightarrow (\varphi \wedge \circ\Box\varphi)$
  - A4: $\circ\neg\varphi \Leftrightarrow \neg\circ\varphi$
  - A5: $\circ(\varphi \Rightarrow \psi) \Rightarrow (\circ\varphi \Rightarrow \circ\psi)$
  - A6: $\Box(\varphi \Rightarrow \circ\varphi) \Rightarrow (\varphi \Rightarrow \Box\varphi)$
  - A7: $\varphi \mathcal{U} \psi \Leftrightarrow (\varphi \wedge \psi) \vee (\varphi \wedge \circ(\varphi \mathcal{V} \psi)$
  - A8: $\varphi \mathcal{U} \psi \Rightarrow \Diamond\psi$
- Result: a sound and relatively complete proof system
- Can implement in Isabelle in much the same way as we did Hoare Logic

Important Meta-Definitions

- A is sound with respect to B if things that are "true" according to A are things that are "true" according to B.
- A is complete with respect to B if things that are "true" according to B are things that are "true" according to A.
- A is sound if things that are "true" according to A are true.
- A is complete everything that is true (that is in the scope of A) is "true" according to A.
- A is relatively complete with repsect to B if A is complete when B is.

Think: A proof system; B mathematical model

Exercise: $(\varphi \wedge \psi) \Rightarrow (\Box\varphi \wedge \Box\psi)$

What is Model Checking?

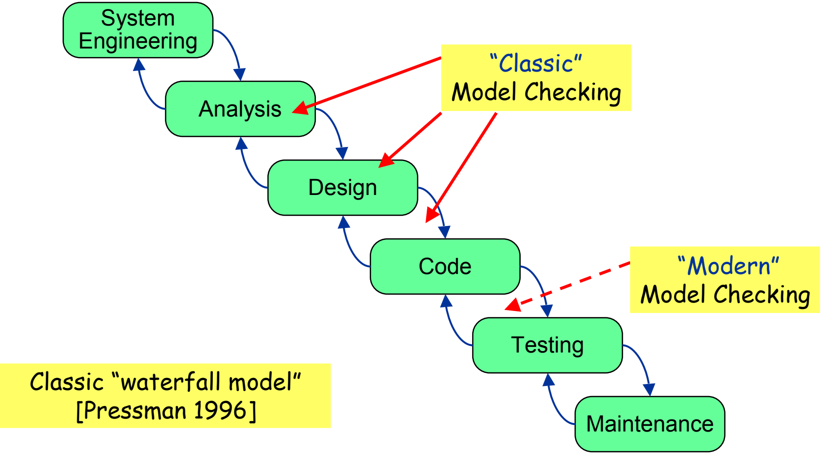
Most generally Model Checking is

- an automated technique, that given
- a finitely presented (think finite-state) model M of a system
- and a logical property φ,
- checks whether the property holds of model: $M \models \varphi$?

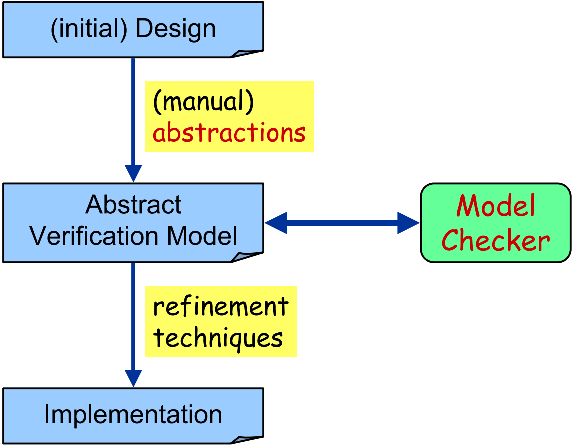
Model Checking

- Model checkers usually give example of failure if $M \not\models \varphi$.
- This makes them useful for debugging.
- Problem: Can only handle finite models: unbounded or continuous data sets can't be directly handled
- Problem: Number of states grows exponentially in the size of the system.
- Answer: Use abstract model of system
- Problem: Relationship of results on abstract model to real system?

System Development



"Classic" Model Checking



LTL Model Checking

- Model Checking Problem: Given model $\mathcal{M}$ amd logical property φ of $\mathcal{M}$, does $\mathcal{M} \models \varphi$?
- Given transition system with states Q, transition relation δ and inital state state I, say $(Q, \delta, I) \models \varphi$ for LTL formula φ if every run of (Q, δ, I), σ satisfies $\sigma \models \varphi$.

Theorem

The Model Checking Problem for finite transition systems and LTL formulae is decideable.

- Treat states $q \in Q$ as letters in an alphabet.
- Language of (Q, δ, I), $\mathcal{L}(Q, \delta, I)$ (or L(Q) for short) is set of runs in Q
- Language of φ, $\mathcal{L}\varphi = \{\sigma | \sigma \models \varphi\}$
- Question: $\mathcal{L}(Q) \subseteq \mathcal{L}(\varphi)$?
- Same as: $\mathcal{L}(Q) \cap \mathcal{L}(\neg\varphi) = \emptyset$?

How to Decide the Model Checking Problem?

- How to answer $\mathcal{L}(Q) \cap \mathcal{L}(\neg\varphi) = \emptyset$?
- Common approach:
  - Build automaton A such the $\mathcal{L}(A) = \mathcal{L}(Q) \cap \mathcal{L}(\neg\varphi)$
  - Are accepting states of A reachable? (Infinitely often?)
- How to build A?
  - One possible answer: Build a series of automata by recursion on structure of $\neg\varphi$.
  - Another possible answer: Build an automaton B such $\mathcal{L}(B) = \mathcal{L}(\neg\varphi)$; take $A = B \times Q$
- Will do at least one approach if time after Spin