EECS 219C: Computer-Aided Verification

Abstraction & Symbolic Model Checking without BDDs

Sanjit A. Seshia

EECS, UC Berkeley

Acknowledgments: Kenneth McMillan

Today's Lecture

- Abstraction
  - Counter-example guided abstraction refinement (CEGAR)
- Symbolic Model Checking without BDDs
  - Uses SAT instead of BDDs
  - Started with Bounded Model Checking
  - Extended to Unbounded Model Checking
    - Abstraction + BMC
    - Interpolation-based model checking (next class)

Abstraction

- Extracting information from a system description that is relevant to proving a property
- Goal: Reduce size of system model
- Terminology:
  - Original model = Concrete system/model

S. A. Seshia 3

Formal Definition

- Abstraction is defined by an abstraction function
- Abstraction function $\alpha : S \rightarrow \hat{S}$
  - S – set of concrete states
  - $\hat{S}$ – set of abstract states
- An abstraction induces an equivalence relation over the concrete states
  - Two concrete states are equivalent if they are mapped to the same abstract state

S. A. Seshia 4

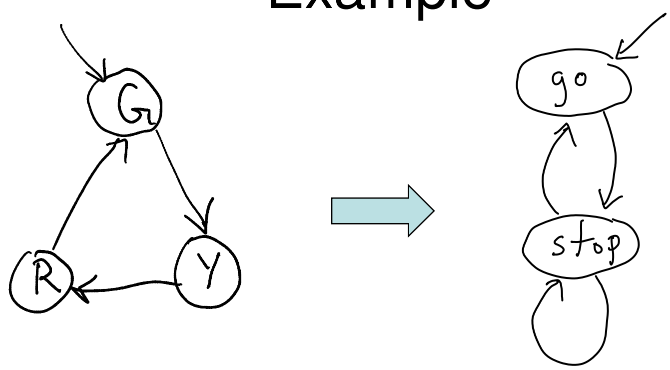
Formal Definition

- Suppose concrete system is (S, S_0, R, L), and abstract system $(\hat{S}, \hat{S}_0, \hat{R}, \hat{L})$
- Abstraction function $\alpha : S \rightarrow \hat{S}$
  - S – set of concrete states
  - $\hat{S}$ – set of abstract states
- $\hat{S}_0 = \{ t \mid \exists s \, . \, S_0(s) \wedge \alpha(s) = t \}$
- $\hat{R} = ?$
  - How do we algorithmically construct $\hat{S}_0$ and $\hat{R}$?
  - How are labels assigned to abstract states?

Example of Abstraction

- Our examples in this lecture will be abstractions that extract a subset of state variables
  - State variables partitioned into: visible and invisible
  - An abstract state is an evaluation of visible variables
  - What is α ?
  - Two concrete states that agree on values of visible variables are grouped together

Example



- Abstractions exhibit more behaviors

Abstraction and Properties

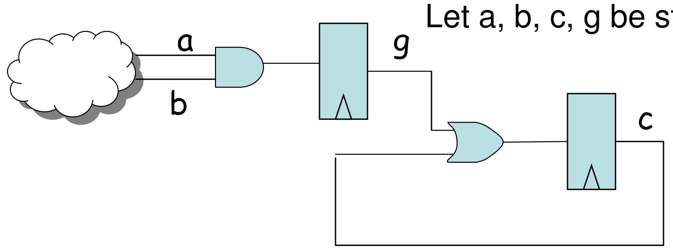
- If an LTL property is true on the abstract model, is it necessarily true on the concrete model?
- If an LTL property is false on the abstract model, is it necessarily false on the concrete model?

Recap: Cone-of-influence

- Suppose the property ϕ mentions a subset of variables V' of the total set V
  - Track variables that V' syntactically depend on, add them to V', and iterate until no new variable dependencies generated
  - Resulting V' is the cone-of-influence and its elements are the visible variables
- Problem: Final V' might be as big as V because it only tracks *syntactic* dependencies
  - But resulting abstraction is precise → if ϕ is false in abstract model it is false in concrete model

Example: Cone-of-influence can be conservative

Let a, b, c, g be state variables



What are the expressions for next state variables c' and g' ?

Suppose we want to prove G(c ➔ Xc) . What's the cone of influence?

If we make g invisible, can we still prove the property?
• what about a and b?

Another approach to Abstraction

- Start with an *arbitrary* subset of variables as visible
  - An option: the ones mentioned in the property
- Construct abstract model, model check it
  - If property passes, we're done
  - If we get a counterexample, check whether it is a counterexample for the concrete model
    - If yes, we're done
    - If not (spurious counterex.) we must make more variables visible and repeat (REFINEMENT)

Counter-Example Guided Abstraction-Refinement (CEGAR)

[R. Kurshan, E. Clarke et al.]

- Start with a choice of α
- Construct abstract model, model check it
  - If property passes, we're done
  - If we get a counterexample, check whether it's is a counterexample for the concrete model (How do we do this?)
    - If yes, we're done
    - If not (spurious counterex.), we must refine α and repeat

Intuition about Refinement

- Remember that α partitions the concrete states into equivalence classes
  - $C_1, C_2, \ldots, C_k$
- A refinement α' can further break up the C_i's
  - States that are equivalent under α' should also be equivalent under α

Formal Definition of Refinement

- α' refines α if
  - $\forall\, s, t \,.\, \alpha'(s) = \alpha'(t) \Rightarrow \alpha(s) = \alpha(t)$
  - $\exists\, s, t \,.\, \alpha'(s) \neq \alpha'(t) \wedge \alpha(s) = \alpha(t)$
- Given above definition, why will the CEGAR iteration terminate?

Visible/Invisible Abstraction

- The set of variables is partitioned into visible V and invisible I
- Questions:
  - How do we construct the abstract model?
    - Given an arbitrary set of visible variables
  - How do we refine the abstraction?
    - i.e., how do we pick new variables to make visible?
    - We want to pick those that will remove the current spurious counterexample

Constructing Abstract Model

- Simply make all invisible variables take arbitrary values
  - Non-deterministically assigned 0 or 1 on each step
- How does this make model checking more efficient?

Constructing Abstract Model

- Simply make all invisible variables take arbitrary values
  - Non-deterministically assigned 0 or 1 on each step
- How does this make model checking more efficient?
  - Avoids some existential quantification, simplifies transition relation

Refining the Abstraction

- The CEGAR approach is most often used today in conjunction with a technique called Bounded Model Checking
- We will study abstraction-refinement in that context

Bounded Model Checking (BMC)

[Biere, Clarke, Cimatti, Zhu, '99]

- Given
  - A FSM M described by S_0, R
  - A property G p and a integer $k \geq 1$
- Determine
  - Does M generate a counterexample to G p of length *k transitions or fewer*?

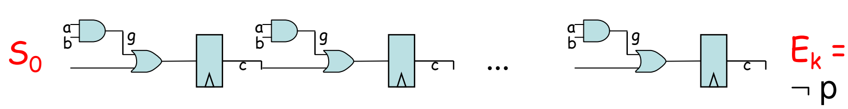
This problem can be translated to a SAT problem. How?

S. A. Seshia 19

Unfolding in BMC

- Unfold the model k times:

  $U_k = R_0 \wedge R_1 \wedge \ldots \wedge R_{k-1}$



- Use SAT solver to check satisfiability of

  $S_0 \;\wedge\; U_k \;\wedge\; E_k$

- A satisfying assignment is a counterexample of k steps

S. A. Seshia 20

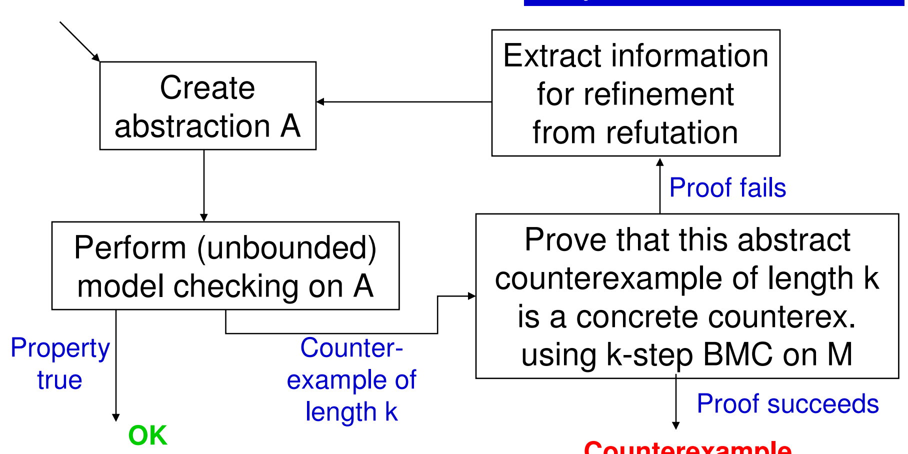
Old view on BMC

- Originally introduced as a debugging tool
  - By finding counterexamples
- Proving properties:
  - Only possible if a bound on the diameter of the state graph is known
    - The diameter is the maximum over shortest path lengths between any two states.
  - Worst case is exponential in system description.

New perspectives: BMC + CEGAR

- BMC + Abstraction can prove properties too!
- Here's how it works:

Why does this terminate?



Steps

1. Create abstraction A ✓
2. Model check A ✓
3. Prove that abstract counterexample is a concrete counterexample using BMC
4. Use refutation of abstract counterexample to do refinement

Checking Abstract Counterex.

- Recall: BMC for length k
  - Use SAT solver to check satisfiability of
    $S_0 \wedge U_k \wedge E_k$
- How do we use this to prove the abstract counterexample of length k also holds for concrete model?

Checking Abstract Counterex.

- Recall: we use BMC for the length k of the abstract counterexample
  - Use SAT solver to check satisfiability of

    $S_0 \wedge U_k \wedge E_k$

    under the partial assignment corresponding to values of the visible variables
  - If SAT solver reports "SAT" we have a concrete counterexample
    - What is a satisfying assignment?
  - If not, we have a refutation ← proof of unsatisfiability

S. A. Seshia 25

Refinement

- Given proof of unsatisfiability of

  $S_0 \wedge U_k \wedge E_k$

  under the partial assignment corresponding to values of the visible variables
- Look at unsatisfiable core of proof
  - Invisible variables that appear in the core indicate why the abstract counterexample is spurious
  - Make those variables visible

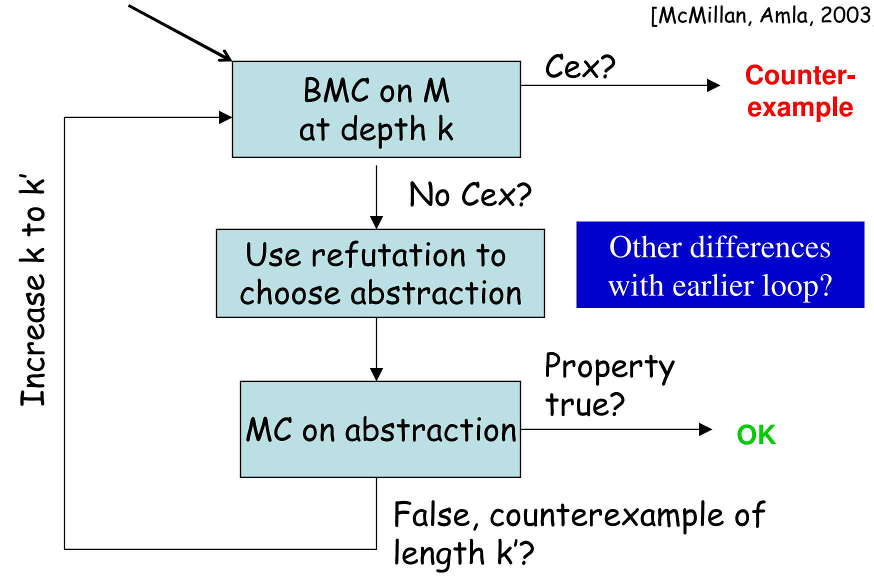
S. A. Seshia 26

Modifying the Abstraction-Refinement Loop

- Insight: Why pick an abstraction to start with?
  - Initial abstraction may not be the best start point
  - Why not do BMC initially and use its results to generate abstractions?

Proof-based Abstraction (PBA)

[McMillan, Amla, 2003]



Termination of PBA

- Depth k increases at each iteration
- Eventually k > diameter d
- If k > d, no counterexample is possible

S. A. Seshia 29

CEGAR vs. PBA

- Refutation via k-step BMC
  - PBA refutes all concrete counterexamples of up to length k
  - CEGAR refutes only the abstract counterexample of length k
- So PBA does more work in the refutation, but usually results in fewer iterations of the loop

S. A. Seshia 30

Next class

- Interpolation-based Model Checking
- Richer kinds of properties (than temporal logic) & verification
  - Mu-calculus, simulation, bisimulation, etc.