Satisfying Error Conditions

Lecture 13
Review

• Verification condition
  \[ \text{precondition} \implies \text{vc( body, postcondition )} \]
  - VC invalid \implies \text{error in program or spec}
  - VC valid \implies \text{program implements spec}

• Error condition
  - negation of VC
  - \text{EC satisfiable} \implies \text{error in program or spec}
  - \text{EC unsatisfiable} \implies \text{program implements spec}

• Satisfiability of ECs?
  - Davis-Putnam
Review: Where Are We?

Program

Specification

\{\}

Semantics

Error

Invalid

VC generation

Verification Condition

negate

Error Condition

satisfiable
Review: Our Specification Language

• Atoms
  - $A ::= E_1 \leq E_2 \mid E_1 = E_2 \mid f(A_1,\ldots,A_n) \mid \ldots$
  - All boolean expressions from our language are atoms
  - Can have an arbitrary collection of predicate symbols
    • reachable($E_1,E_2$) - list cell $E_2$ is reachable from $E_1$

• Literals
  - $L ::= A \mid \neg A$

• Formulas
  - $P ::= L \mid true \mid false \mid P_1 \land P_2 \mid P_1 \lor P_2 \mid \neg P$
Strategy for deciding satisfiability of ECs

• Leverage SAT solver to reason about boolean structure

• Literals?
  - eg \( x < 3 \; x > y \; y > 4 \)
  - use separate routine “checkSatLits(...)” to reason about satisfiability of conjunctions of literals

• Get SAT solver and checkSatLits(...) to cooperate to decide satisfiability of EC formulas
Satisfiability of ECs: Example

- **EC**
  \[(a=b) \land (\neg(f(a)=f(b)) \lor b=c) \land \neg(f(a)=f(c))\]

- Use {...} as funny syntax for boolean variables

- **SAT problem**
  \[
  \begin{aligned}
  &\{a=b\} \land (\neg\{f(a)=f(b)\} \lor \{b=c\}) \land \neg\{f(a)=f(c)\} \\
  &v1 \land (\neg \ v2 \lor \ v3 ) \land \neg \ v4
  \end{aligned}
  \]

- **SAT solution**
  - True: \( \{a=b\} \)
  - False: \( \{f(a)=f(b)\} \ \{b=c\} \ \{f(a)=f(c)\} \)

- **Conjunction of literals**
  \[a=b \land f(a)\neq f(b) \land b\neq c \land f(a)\neq f(c)\]

- Is this conjunction of literals satisfiable?
Satisfiability of ECs: Example

• Conjunction of literals
  \[ a=b \land f(a) \neq f(b) \land b \neq c \land f(a) \neq f(c) \]

• Is this conjunction of literals satisfiable?
  - simpler problem, no disjunctions, no case analysis
  - call routine checkSatLits(L_1, ..., L_k)
  - returns no!

• Semantics of equality says:
  - \( a=b \Rightarrow f(a)=f(b) \)
  - this tautology returned by checkSatLits(L_1, ..., L_k)
  - append tautology to original problem, and repeat!
Satisfiability of ECs: Example

• EC + tautology
  - \((a=b) \land (\neg f(a)=f(b)) \lor b=c \land \neg (f(a)=f(c)) \land (a=b \Rightarrow f(a)=f(b))\)

• SAT problem
  - \({a=b} \land (\neg \{f(a)=f(b)\} \lor \{b=c\}) \land \neg \{f(a)=f(c)\} \land (\{a=b\} \Rightarrow \{f(a)=f(b)\})\)

• SAT solution
  - True: \({a=b} \quad \{f(a)=f(b)\} \quad \{b=c\}
  - False: \{f(a)=f(c)\}

• Conjunction of literals
  - \(a=b \land f(a)=f(b) \land b=c \land f(a) \neq f(c)\)

• Is this conjunction of literals satisfiable?
Satisfiability of ECs: Example

• Conjunction of literals
  - $a = b \land f(a) = f(b) \land b = c \land f(a) \neq f(c)$

• Is this conjunction of literals satisfiable?
  - call routine checkSatLits($L_1, ..., L_k$)
  - returns no!

• Semantics of equality says:
  - $a = b \land b = c \Rightarrow f(a) = f(c)$
  - this tautology returned by checkSatLits($L_1, ..., L_k$)
  - append tautology to original problem, and repeat!
Satisfiability of ECs: Example

• EC + tautologies
  - \((a=b) \land (\neg (f(a)=f(b)) \lor b=c) \land \neg (f(a)=f(c))\)
  - \((a=b \Rightarrow f(a)=f(b))\)
  - \((a=b \land b=c \Rightarrow f(a)=f(c))\)

• SAT problem
  - \(\{a=b\} \land (\neg \{f(a)=f(b)\} \lor \{b=c\}) \land \neg \{f(a)=f(c)\}\)
  - \(\{a=b\} \Rightarrow \{f(a)=f(b)\}\)
  - \(\{a=b\} \land \{b=c\} \Rightarrow \{f(a)=f(c)\}\)

• Unsatisfiable
  - therefore original EC unsatisfiable
  - we’re done!
  - leverages SAT solver and checkSatLits\((L_1, ..., L_k)\)
Review: Where Are We?

Program \{ Specification \} \rightarrow Semantics \rightarrow Error

- Invalid
- VC generation

Verification Condition

Error Condition

check satisfiability using Davis-Putnam and checkSatLits(…)

negate
Implementing checkSatLits($L_1, \ldots, L_k$)

- A **theory** consists of:
  - A set of function and predicate symbols (syntax)
  - Definitions for the meaning of these symbols (semantics)
    - Semantic or axiomatic definitions

- Example:
  - Symbols: $0, 1, -1, 2, -2, \ldots, +, -, =, <$ (with the usual meaning)
    - Theory of integers with arithmetic (Presburger arithmetic)
    - Satisfiable? $y > 2x + 1 \land y + x > 1 \land y < 0$

- The **Satisfiability Problem**: Decide whether a conjunction of literals in the theory is satisfiable
Examples of Theories. Equality.

• The theory of equality with uninterpreted functions
• Symbols: =, ≠, f, g, ...
• Axiomatically defined:

\[
\begin{align*}
E &= E \\
E_1 &= E_2 \\
E_1 &= E_2 & E_2 &= E_3 \\
E_1 &= E_3 \\
f(E_1) &= f(E_2)
\end{align*}
\]

• Example of a satisfiability problem:

\[
g(g(g(x))) = x \land g(g(g(g(g(x)))))) = x \land g(x) \neq x
\]
A Satisfiability Procedure for Equality

• Let $R$ be a relation on terms

• The \textbf{equivalence closure} of $R$ is the smallest relation closed under reflexivity, symmetry and transitivity
  - an \textit{equivalence relation}
  - divides terms into \textit{equivalence classes}

• \textbf{Computing the equivalence closure}
  - each equivalence class has a \textit{representative element}
  - given a term $t$ we say that $t^*$ is its \textit{representative element}
  - two terms $t_1$ and $t_2$ are equivalent iff $t_1^* = t_2^*$
  - computable in near-linear time (union-find)
A Satisfiability Procedure for Equality (Cont)

• Let $R$ be a relation on terms

• The **congruence closure** of $R$ is the smallest relation that is closed under equivalence (reflexivity, symmetry and transitivity) and congruence
  - also an *equivalence relation*
  - divides term in *equivalence classes*

• Computing the congruence closure
A Representation for Symbolic Terms

- We represent terms as DAGs
  - Share common subexpressions
  - E.g. $f(f(a, b), b)$:

- Equalities are represented as dotted edges
  - E.g. $f(f(a, b), b) = a$
  - We consider the transitive closure of dotted edges
Computing Congruence Closure

- We pick arbitrary representatives for all equivalence classes (nodes connected by dotted edges)

- For all nodes \( t = f(t_1, ..., t_n) \) and \( s = f(s_1, ..., s_n) \)
  - If \( t_i^* = s_i^* \) for all \( i = 1..n \) (find)
  - We add an edge between \( t^* \) and \( s^* \) and pick one of them as the representative for the entire class (union)
Computing Congruence Closure (Cont.)

• Congruence closure is an inference procedure for the theory of equality
  - Always terminates because it does not add nodes

• The hard part is to detect the congruent pairs or terms
  - There are tricks to do this in $O(n \log n)$

• We say that $f(t_1, ..., t_n)$ is represented in the DAG if there is a node $f(s_1, ..., s_n)$ such that $s_i^* = t_i^*$
Satisfiability Procedure for Equality

1. Given \( F = \bigwedge_i t_i = t_i' \land \bigwedge_j u_j \neq u_j' \)
2. Represent all terms in the same DAG
3. Add dotted edges for \( t_i = t_i' \)
4. Construct the congruence closure of those edges
5. Check that for all \( j \) we have \( u_j^* \neq u_j'^* \)

**Theorem:**

\( F \) is satisfiable if and only if for all \( j \) \( u_j^* \neq u_j'^* \)
Example with Congruence Closure

- Consider: $g(g(g(x))) = x \land g(g(g(g(g(x)))))) = x \land g(x) \neq x$