Satisfying Error Conditions 3

Lecture 15
Overview of Verification Architecture

Program

Specification

Verification Condition

Error Condition

negate

checkSatLits

checkSatLitsArith

checkSatLitsEquality

checkSatLitsArrays

EC satisfiability checker

SAT solver

Davis-Putnam

conjunction of literals

Nelson-Oppen cooperating decision procedures
## EC Satisfiability Checker

<table>
<thead>
<tr>
<th>{a=b}</th>
<th>{f(a)=f(b)}</th>
<th>{a=c}</th>
<th>{f(a)=f(c)}</th>
<th>Sat?</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>X 0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>X 0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

The implication \((a=b \Rightarrow f(a)=f(b))\) is an explicaded tautology, which removes many other truth assignments.

\[
\begin{align*}
a = b \\
\land f(a) \neq f(b) \\
\land b \neq c \\
\land f(a) \neq f(c)
\end{align*}
\]
Overview of Verification Architecture

Program

Specification

Program

Specification

Verification Condition

Sat solver

Davis-Putnam

EC

satisfiability

checker

SAT solver

Davis-Putnam

checkSatLits

conjunction of literals

checkSatLitsArith

checkSatLitsEquality

checkSatLitsArrays

Error Condition

negate

Nelson-Oppen cooperating decision procedures
checkSatLitsArith: Difference Constraints

• A special case of linear arithmetic
• All constraints of the form:
  \( x + c \leq y \)
• \( c \) is a constant
• Special variable \( z \) representing 0

• Example
  - \( x \leq y \)
  - \( y+4 \leq w \)
  - \( w-2 \leq x \)
  - \( w+1 \leq z \)
checkSatLits Equality

- Consider: \( g(g(g(x))) = x \land g(g(g(g(g(x)))))) = x \land g(x) \neq x \)
Overview of Verification Architecture

Program \{ Specification \} \rightarrow Verification Condition \rightarrow Error Condition

- SAT solver
  - Davis-Putnam
- EC
  - satisfiability checker
- checkSatLits
- checkSatLitsArith
- checkSatLitsEquality
- checkSatLitsArrays
- Nelson-Oppen cooperating decision procedures
- conjunction of literals
Cooperating Satisfiability Procedures

- Consider equality and arithmetic

\[
f(f(x) - f(y)) \neq f(z) \quad x \leq y \quad y + z \leq x \quad 0 \leq z
\]

\[
f(x) = f(y) \quad x = y
\]

\[
f(x) - f(y) = z
\]

\[
0 = z
\]

false \quad f(f(x) - f(y)) = f(z)
3. Broadcast all discovered equalities and re-run sat. procedures
   • Until no more equalities are discovered or a contradiction arises
Overview of Verification Architecture

Program

Specification

Verification Condition

Error Condition

classification

negate

checkSatLitsArith

checkSatLitsEquality

checkSatLitsArrays

SAT solver

Davis-Putnam

EC satisfiability checker

checkSatLits

conjunction of literals

Nelson-Oppen cooperating decision procedures

checkSatLitsArrays

checkSatLitsArith

checkSatLitsEquality

checkSatLitsArrays
Theory of Arrays

- Syntax and informal semantics:
  - If $E$ denotes an address and $\mu$ a heap state then:
  - $\text{sel}(\mu, E)$ denotes the contents of memory cell
  - $\text{upd}(\mu, E, V)$ denotes a new heap state obtained from $\mu$ by writing $V$ at address $E$

- Decision procedure implements following rule
  \[
  x = y \Rightarrow \text{sel}(\text{upd}(\mu, x, v), y) = v \\
  x \neq y \Rightarrow \text{sel}(\text{upd}(\mu, x, v), y) = \text{sel}(\mu, y)
  \]
  - what if $x = y$ is unknown?
Theory of Arrays

• Syntax and informal semantics:
  - If $E$ denotes an address and $\mu$ a heap state then:
  - $sel(\mu, E)$ denotes the contents of memory cell
  - $upd(\mu, E, V)$ denotes a new heap state obtained from $\mu$ by writing $V$ at address $E$

• Decision procedure implements following rule

  $x \neq y \lor sel(upd(\mu, x, v), y) = v$
  $x = y \lor sel(upd(\mu, x, v), y) = sel(\mu, y)$

  - what if $x = y$ is unknown?
  - *non-convex theory:* input facts entail disjunction of equalities, but do not entail any individual equality
  - can add SAT literal $\{x=y\}$
Overview of Verification Architecture

Program

Specification

Verification Condition

Error Condition

EC
satisfiability checker

SAT solver
Davis-Putnam

checkSatLits

checkSatLitsArith

checkSatLitsEquality

checkSatLitsArrays

 negate

conjunction of literals

Nelson-Oppen cooperating decision procedures
Example

//@ requires x != y
//@ ensures *x < *y+1

void sort2(int *x, int *y) {
    if (*x > *y) {
        int t = *x;
        *x = *y
        *y = t;
    }
}

• Class Challenge: Use theorem proving techniques to cooperatively verify the correctness of sort2