Symbolic Exploration

Azadeh Farzan
CS410 - Fall 2020
Reachability

One of the **simplest** verification problems:

- Given a **set of bad states**, can a program/system **reach** one of these states during its execution?
- It is a **decision problem**.
Propose a trivial algorithm for reachability...
Depth First Search!
State Space Exploration

A program with 100 Boolean variables can have up to $2^{100}$ different reachable states.

Representing these individually is not feasible.

Symbolic representation accommodates representing sets of states more compactly.

$p, q, r : \overline{p} \quad q \land \lnot r$

4 states \quad 2 states
Formally ...

A boolean formula $F$ represents the set of all states $s$ where $s \models F$

$$\text{states}(F) = \{ s \mid s \models F \}$$
Logical Connectives as Set Operations

Let $S_1 = \text{states}(F_1)$ and $S_2 = \text{states}(F_2)$

$S_1 \cup S_2 = \text{states}(F_1 \lor F_2)$

$S_1 \cap S_2 = \text{states}(F_1 \land F_2)$

$\overline{S_1} = \text{states}(\neg F_1)$
How do we solve reachability symbolically?
$\text{step}(\vec{v}, \vec{v}')$: a formula over two copies of state
$R_0 = I \rightarrow set \ of \ initial \ states$

$R_1 (\vec{v}) = [R_0 (\vec{v}') \land step (\vec{v}', \vec{v})] \lor R_0 (\vec{v})$

reachable by one step from $R_0$

$R_2 (\vec{v}) = [R_1 (\vec{v}) \land step (\vec{v}; \vec{v})] \lor R_1 (\vec{v})$

$\vdots$
Let $E(\mathfrak{F})$ represent all error states.

At each step $j$, if $R_j(\mathfrak{F}) \land E(\mathfrak{F})$ is satisfiable, then an error state is reachable.
\[ R_0 = 1, R_1, R_2, R_3, \ldots \]

If the system is finite-state then

\[ \exists j : R_j = R_{j+1}. \]

We either find a \( j \) s.t. \( R_j \wedge E \) is satisfiable or a \( j \) s.t. \( R_j = R_{j+1}. \)
Let's build a better reachability algorithm!
Property-Directed Reachability
Setup

Clause: disjunction of literals

Cube: conjunction of literals

Each frame $R_j$ is a CNF formula.

But now, it is an over-approximation of the set of reachable states in $j$ steps.

$CL(F)$: set of clauses in CNF formula $F$. 
Invariants

- $R_0 = I$
- $R_j \subseteq R_{j+1}$
- $\text{CL}(R_{j+1}) \subseteq \text{CL}(R_j)$
- $T(R_j) \subseteq R_{j+1}$ \( T: \text{short for step} \)
- $R_j \subseteq \neg E$ \( \text{except the last frame } N \)
frames

$R_0 \subset R_1 \subset R_2 \subset \ldots \subset R_N \subset R_{	ext{ntl}}$

$R_{	ext{ntl}}$ is not SAT
The Algorithm

Check if $R_N \land E$ is SAT.

• No? $R_N \subseteq \neg E$
  • new empty frame $R_{N+1}$
  • $\forall j > 0$, push clauses from $R_j$ to $R_{j+1}$
    • clause $c \in CL(R_j)$ can be pushed if $R_j \land A \land \neg C'$ is not SAT.
  • Terminate if two equal frames found
• Is $s$ truly reachable? Yes!
• Is $t$ reachable from the level before?
• Is $s$ truly reachable? **No!**
• $s$ is a cube $\implies \neg s$ is a clause
The Algorithm

- **Check if** $R_n \not\models \phi$ **is SAT.**

  - **Yes?** → *careful: $R_n$ was overshooting!*
    - There is a satisfying assignment $s$.
  - **Check if** $R_{n-1} \models \phi s'$ **is SAT**
    - **NO?** Add $\neg s$ to $R_n$ and start over
    - **YES?** get assignment $\bar{t}$
      - repeat step (* with $(R_{n-2}, \bar{t})$
Wrap up