Test Case Generation for Ultimately Periodic Paths

Saddek Bensalem, Doron Peled, Hongyang Qu, Stavros Tripakis, Lenore Zuck

Haifa Verification Conference, 2007
Outline

- Background (flow charts, path preconditions)
- Conditions for ultimately periodic paths
- Test case generation methodology
- Implementation
- Conclusion
Flow Charts

- A graphical representation of structure of a program
- Three kinds of nodes
  - Ellipse (beginning, end)
  - Box (assignment)
  - Diamond (condition)
- Two kinds of edges
  - Outgoing from ellipse or box nodes (no labels)
  - Outgoing from diamond nodes (labelled as yes or no)
An example

- **Program 1**
  ```
  while (x<=y && z>0) {
    y := y / 2;
    x := x * 2;
    z := z - 1;
  }
  ```

- **Program 2**
  ```
  while (x>=y) {
    x := x - 1;
    y := y + 1;
    z := z * 2
  }
  ```
Path conditions

- A path condition $\varphi_\mu(\varphi)$ is a first order predicate that expresses the condition to execute the path $\mu$ and satisfy the predicate $\varphi$ at the end of the execution.

- Sometime we write $\varphi_\mu$ for $\varphi_\mu(\text{true})$. 
Computing path conditions

1. \( \varphi \land c \)
   - yes \( \varphi \)
   - \( \varphi \land \neg c \)

2. \( \varphi \land c \)
   - no \( \varphi \)

3. \( \varphi[e/x] \)
   - \( x := e \)

\[ \varphi_\mu = y/2 \leq x \leq y \land z > 0 \]

- yes \( x \geq y/2 \)
- \( y := y/2 \)
- \( x \geq y \)
- true
- yes
- \( x := x - 1 \)

- \( \varphi = true \)
Properties of path conditions

- **Compositionality**
  \[ \phi_{\sigma \rho} (\varphi) = \phi_{\sigma} (\phi_{\rho} (\varphi)) \]

- **Distribution over conjunction**
  \[ \phi_{\mu} (\varphi \land \psi) = \phi_{\mu} (\varphi) \land \phi_{\mu} (\psi) \]

- **Monotonicity**
  if \( \varphi \rightarrow \psi \) then \[ \phi_{\mu} (\varphi) \rightarrow \phi_{\mu} (\psi) \]
How to calculate a path condition for an ultimately periodic path?

- This is the subject of this work.
- In general this is an undecidable problem.
Some conditions for ultimately periodic paths

- **Equality condition**
  computed using equality method

- **Monotonicity condition**
  computed using monotonicity method

- **Condition for not completely ultimately periodic paths**
Equality method

- We are looking for the condition to execute a loop $\rho$ indefinitely, after a finite prefix $\sigma$, where in each iteration, the variables obtain the same values.

- Executing the periodic part $\rho$ once when $\varphi_\rho \land X = tr_\rho(X)$.

- Executing it after the prefix $\sigma$ is when $\varphi_{\sigma\rho} \land \varphi_\sigma(\varphi_\rho \land X = tr_\rho(X))$.

- Simplifying: $\varphi_{\sigma\rho} \land \varphi_\sigma(X = tr_\rho(X))$. 
**Example (1)**

\[ x > 0 \]

\[ z := z - 1 \]

\[ x > 0 \]

Yes

\[ y := x \]

\[ x := \frac{2x + y + z}{3} \]

\[ x > 0 \]

\[ x = \frac{3x + z}{3} \Rightarrow z = 0 \]

\( (x, y, z) \)

\[ y = x \Rightarrow y = x \]

\[ \phi_{\sigma}(z = 0 \land y = x) \equiv z - 1 = 0 \land y = x \]

\[ \phi_{\rho \rho}(true) \equiv x > 0 \]

\[ x > 0 \land z - 1 = 0 \land y = x \]

\[ \left\{ \frac{3x + z}{3}, x, z \right\} \]
Monotonicity Method

- It is sufficient to find a loop invariant such that $I \rightarrow \varphi_\rho(I)$.
- The weakest such invariant $I$ is $I = \varphi_\rho(true)$.
- Proof:
  $I \rightarrow true$ for each $I$.
  By monotonicity of $\varphi$, $\varphi_\rho(I) \rightarrow \varphi_\rho(true)$.
  Since $I \rightarrow \varphi_\rho(I)$, it holds that $I \rightarrow \varphi_\rho(true)$, independently of $I$. 
Deriving an ultimately periodic condition

- We set $I = \varphi_{\rho}(true)$ in the implication $I \rightarrow \varphi_{\rho}(I)$, obtaining $\varphi_{\rho}(true) \rightarrow \varphi_{\rho}(\varphi_{\rho}(true))$.

- This can be rewritten as $\varphi_{\rho}(true) \rightarrow \varphi_{\rho}(true)[tr_{\rho}(X)/X]$.

- Applying the $\varphi$ of the prefix, we obtain $\varphi_{\sigma}(\varphi_{\rho}(true)) \rightarrow \varphi_{\sigma}(\varphi_{\rho}(true)[tr_{\rho}(X)/X])$.

- The next slide will deal with the 2nd bullet (and then we need to remember to apply the 3rd).
The case where $\mathcal{E}_\rho(\text{true})$ is $e \geq 0$ (or $e > 0$)

- Set $e' = e[tr_\rho(X) / X]$.
- Bullet 2 from previous slide becomes $e \geq 0 \rightarrow e' \geq 0$.
- A sufficient condition is $e' \geq e$.
- Other cases: when we have a condition $\mathcal{E}_\rho(\text{true}) \equiv g \geq f$, we take $e = g - f$.
- **Conjunction principle:** In case $\mathcal{E}_\rho(\text{true}) \equiv g \geq 0 \land f \geq 0$, we have condition $g' \geq g \land f' \geq f$.
- **Disjunction principle:** In case $\mathcal{E}_\rho(\text{true}) \equiv g \geq 0 \lor f \geq 0$, it is sufficient that we strengthen to either $g' \geq g$ or $f' \geq f$.
- An equality can be transformed into two inequalities and the disjunction case is applied.
Example (2)

\( x > 0 \)

\[ z := z - 1 \]

\( x > 0 \)

\( (x, y, z) \)

\( \phi_\rho (true) \equiv x > 0 \Rightarrow \frac{3x + z}{3} > x \Rightarrow z > 0 \)

\( \phi_\sigma (\omega_\rho) \rightarrow \phi_\sigma (\phi_\rho [tr_\rho (X) / X]) \equiv x > 0 \land z > 1 \)

\( \frac{3x + z}{3}, x, z \)
Some mixed and not completely ultimately periodic paths

While x>1 do
begin
if PowerTwo(x-1) then
    x:=4*(x-1)
else
    x:=x-1
end.

Example: 4→3→8→7→6→5→16→15…
Computing the condition

- Shrinking the loop body to a new transition $t$:
  \[ \wp_{\sigma_t}(\varphi) = \bigvee_i (c_i \land \wp_{\sigma}(\varphi)[\bar{e}_i / \bar{x}_i]) \]

- Example:
  
  $t : \text{PowerTwo}(x - 1) \mapsto x := 4(x - 1) \oplus \neg \text{PowerTwo}(x - 1) \mapsto x := x - 1$

  \[ \wp_{\sigma_t} = (\text{PowerTwo}(x - 1) \land 4(x - 1) > 1) \lor (\neg \text{PowerTwo}(x - 1) \land x - 1 > 1) \]

  \[ \wp_{\sigma_t} \rightarrow x > 1 \]
Test case generation

- LTL $\rightarrow$ Automaton
- Compiler
- Flow chart
- Model Checker
- Path
- Path condition calculation
- Transitions
- First order instantiator
- Test monitoring
Goals

- Verification of software.
- Compositional verification. Use only a unit of code instead of the whole code.
- Parameterized verification. Verifies a procedure with any value of parameters in “one shot”
- Generating test cases via path conditions: A truth assignment satisfying the path condition. Helps derive the demonstration of errors.
- Generating appropriate values to missing parameters.

24/10/2007 Hongyang Qu, Imperial College London
How to generate test cases

- Take the intersection of an LTL automaton with the flow graph. Some paths would be eliminated for not satisfying the assertions on the program counters.
- Seeing same flow chart node does not mean a loop: program variables may value. Use iterative deepening.
- For each initial path calculate the path condition. Backtrack if condition simplifies to false.
- Report path condition based on flow graph path+LTL assertions.
- Always simplify conditions!
intersection of the property automaton and flow graph

\[ x \geq y \]

\[ x \geq 2 \times y \]
How the LTL formula directs the search

- **Spec:** \((x = 4)U(x = 5 \land O...)\)
Implementation

- Implemented in Java
- Using *Mathematica* to simplify conditions.
- Detecting identical states
- Heuristic match
Conclusion

- An approach for generating test cases automatically.
- Also: verification of infinite state systems.
- Path by path verification rather than state by state.
- Challenge: the weakest precondition for ultimately periodic sequences in infinite state systems.
- We suggested several methods (e.g., the equality and monotonicity methods, etc.)
- Not all of the infinite executions are ultimately periodic.