Temporal Logic Based Approach to Test Coverage and Generation

Hyoung S. Hong, Insup Lee, Oleg Sokolsky

SDRL (Systems Design Research Lab)
RTG (Real-Time Systems Group)
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA
Software Development Process

- **Requirements capture and analysis**
  - Informal to formal
  - Consistency and completeness
  - Assumptions and interfaces between system components
  - Application-specific properties
- **Design specifications and analysis**
  - Formal modeling notations
  - Abstractions
  - Analysis techniques (simulation, model checking, equivalence checking, testing, etc.)
- **Implementation**
  - Manual/automatic code generation
  - Validation (testing, model extraction, etc.)
  - Run-time monitoring and checking
Overall Structure
Specification-Based Testing

• Determines whether an implementation conforms to its specification
  – Hardware and protocol conformance testing
  – Widely-used specifications
    • Finite state machines and labeled transition systems

• Two main steps
  – Test generation from specifications
    • What to test, how to generate test
  – Test execution of implementations
    • Applies tests to implementations and validates the observed behaviors
Our Approach

• The problem: automatic test generation from specifications
  – Specifications: EFSMs, Flow graph
  – Coverage criteria
    • Control flow: all-states, all-transitions, etc.
    • Data flow: all-defs, all-uses, all-inputs, all-outputs, etc.

• Test generation

\[\text{Specification} \rightarrow \text{Input to model checker} \rightarrow \text{Model checker} \rightarrow \text{A set of tests}\]

\[\text{Coverage criterion} \rightarrow \text{A set of formulas}\]
Specifications: EFSM

\[ \langle S, S_0, E, V, T \rangle \]
- \( S \): a set of states
- \( S_0 \): initial state
- \( E \): a set of input/output events
- \( V \): variables are manipulated by transitions
- \( T \): a set of transitions

\[ t1: \text{insert}[m+x \leq 5] \]
\[ /m := m + x \]

\[ t2: \text{coffee}[m > 1] \]
\[ /m := m - 1 \]

\[ t3: \text{done} \]

\[ t4: \text{display}/y := m \]

\[ t5: \text{display}/y := m \]
Coverage criteria in WCTL

- Each coverage criterion is represented by a set of temporal logic formulas
  - WCTL: a subset of CTL
    - Atomic propositions $p_1, ..., p_n$
    - Temporal operators $\text{EX}$, $\text{EU}$, $\text{EF}$
    - Conjunctions have at most one non-atomic conjuncts
    - Negations can be applied only to atomic propositions
    - Unrestricted disjunctions
    - E.g.: $\text{EF}(p_1 \& \text{EF}p_2)$
  - WCTL formulas have linear witnesses
All-states coverage criterion

- Requires every state be covered at least once
- With every state $s$, associate $EF(s \land EF_{exit})$

### Example

- **t1**: insert $[m+x \leq 5]$
  - $/m := m+x$
- **t2**: coffee $[m > 1]$
  - $/m := m-1$
- **t3**: done
- **t4**: display $/y := m$
- **t5**: display $/y := m$

### Equations
- $EF(idle \land EF_{exit})$
- $EF(busy \land EF_{exit})$
All-transitions coverage criterion

- Requires every transition be covered at least once
- With every transition $t$, associate $EF(t \& EF_{exit})$

$\begin{align*}
\text{t1: insert}[m+x\leq 5] & \quad \text{EF}(t1 \& EF_{exit}) \\
/m:=m+x & \\
\text{t2: coffee}[m>1] & \quad \text{EF}(t2 \& EF_{exit}) \\
/m:=m-1 & \\
\text{t3: done} & \quad \text{EF}(t3 \& EF_{exit}) \\
\text{t4: display}/y:=m & \quad \text{EF}(t4 \& EF_{exit}) \\
\text{t5: display}/y:=m & \quad \text{EF}(t5 \& EF_{exit})
\end{align*}$
Data flow: definitions and uses

- **Central notions in the data-flow analysis**
- **Definition:** a value is assigned to a variable
- **Use:** a value of a variable is used in an expression
- **Variables are defined and used in transitions**
- **Definition-use pair:** \((v,t,t')\)
  - \(v\) is defined by \(t\)
  - \(v\) is used by \(t'\)
  - There is a path from \(t\) to \(t'\) free from other definitions of \(v\)
Covering a definition-use pair

- With a definition-use pair \((v, t, t')\), associate
  - \(\text{EF}(t \& \text{EXE}[!\text{def}(v) \cup (t' \& \text{EF}exit)])\)
  - \(\text{def}(v)\): disjunction of all transitions that define \(v\)

\[
\begin{align*}
t1: & \text{ insert}[m+x \leq 5] \\
 & /m := m + x
\end{align*}
\]

\[
\begin{align*}
t2: & \text{ coffee}[m > 1] \\
 & /m := m - 1
\end{align*}
\]

\[
\begin{align*}
t3: & \text{ done}
\end{align*}
\]

\[
\begin{align*}
t4: & \text{ display}/y := m
\end{align*}
\]

\[
\begin{align*}
t5: & \text{ display}/y := m
\end{align*}
\]
Data-flow coverage criteria

- **All-defs coverage criterion**
  - Requires a definition-clear path from *every* definition to *some* use be covered at least once

- **All-uses coverage criterion**
  - Requires a definition-clear path from *every* definition to *every* use be covered at least once

```
# Transition System

01: insert [m + x <= 5]
   /m := m + x

02: coffee [m > 1]
   /m := m - 1

03: done

04: display /y := m

05: display /y := m
```

**All-uses coverage criterion**

\[
\begin{align*}
&EF(t_1 \land \text{EXE}[(\neg t_1 \lor t_2) \lor (t_1 \land \text{EF} \text{exit})]) \\
&EF(t_1 \land \text{EXE}[(\neg t_1 \lor t_2) \lor (t_2 \land \text{EF} \text{exit})]) \\
&EF(t_1 \land \text{EXE}[(\neg t_1 \lor t_2) \lor (t_4 \land \text{EF} \text{exit})]) \\
&EF(t_1 \land \text{EXE}[(\neg t_1 \lor t_2) \lor (t_5 \land \text{EF} \text{exit})]) \\
&EF(t_2 \land \text{EXE}[(\neg t_1 \lor t_2) \lor (t_1 \land \text{EF} \text{exit})]) \\
&EF(t_2 \land \text{EXE}[(\neg t_1 \lor t_2) \lor (t_2 \land \text{EF} \text{exit})]) \\
&EF(t_2 \land \text{EXE}[(\neg t_1 \lor t_2) \lor (t_4 \land \text{EF} \text{exit})]) \\
&EF(t_2 \land \text{EXE}[(\neg t_1 \lor t_2) \lor (t_5 \land \text{EF} \text{exit})])
\end{align*}
\]
Data flow chains

• **Affect pair** \((v, t, v', t')\)
  - value of \(v\) used by \(t\) affects the value of \(v'\) defined at \(t'\)
    - \((v, t)\) directly affects \((v', t)\)
    - Either \((v, t)\) directly affects \((v', t')\) or there is a definition-use pair \((v'', t, t'')\) such that \((v, t)\) directly affects \((v'', t)\) and \((v'', t'')\) affects \((v', t')\)

\[\begin{align*}
  t_1 &: \text{insert}[m+x \leq 5] \\
       &: /m := m + x
\end{align*}\]

\[\begin{align*}
  t_2 &: \text{coffee}[m > 1] \\
       &: /m := m - 1
\end{align*}\]

\[\begin{align*}
  t_3 &: \text{done}
\end{align*}\]

\[\begin{align*}
  t_4 &: \text{display}/y := m
\end{align*}\]

\[\begin{align*}
  t_5 &: \text{display}/y := m
\end{align*}\]

\( (x, t_1) \) directly affects \((m, t_1)\)

\( (x, t_1) \) affects \((y, t_5)\)

4/10/02 TACAS'02
Data flow chain coverage

• **Affect pair** \((v, t, v', t')\)
  – May consist of an arbitrary number of definition-use pairs
  – *We extend CTL with least fixpoint operators*
    • Alternatively, we can use (alternation-free) mu-calculus

• **All-inputs coverage criterion**
  – Requires a path from **every** input to **some** output be covered at least once

• **All-outputs coverage criterion**
  – Requires a path from **every** input to **every** output be covered at least once
Witness Generation

• Generating a witness for a formula
  – Cost: the length of a witness
  – A minimal-cost witness for a formula
    • Existing model checkers generate a minimal-cost witness by breadth-first search of state space

\[ E[\emptyset U \bullet \bullet] \]
Test Generation

- **A set of witnesses for a set of formulas**
  - **Costs**
    - The total length of witnesses or
    - The number of witnesses (reset operation is expensive)
  - Both optimization problems are NP-hard (Hitting Set Problem)

\[
\begin{align*}
E[\bigcup] & \quad E[\bigcup] \\
E[\bigcup U \bigcup] & \\
E[\bigcup U \bigcup] & \\
E[\bigcup U \bigcup] & \\
\end{align*}
\]
Model-based test generation

• Testing remains the primary validation technique
• Model-based test generation adds rigor to testing:
  - Provide test suits based on a formally verified model
  - Conventional testing coverage criteria applied to the model
• Developed a framework for test generation:
  - Model is Extended Finite-State Machines (EFSM)
  - Offers control-flow (e.g., state coverage, transition coverage) and data-flow (e.g., all-def, all-use coverage) criteria
• Test generation from formal specifications
  - Hierarchical reactive modules
    • EFSM + hierarchy + concurrency
  - Hybrid systems
    • CHARON: EFSM + hierarchy + concurrency + differential equations
• Case studies and tool development
  - Infusion pump system, SONY Aibo, FDA bloodbank policy
  - Application to certification process
• Basis for conformance metrics
Current Work

- A unified framework for test generation
  - Test generation is model checking of abstractions

Some abstract interpretations:
- Flow graphs
- Abstraction graphs
- Reachability graphs

Related works:
- ICSE '03: Fully syntactic
- TACAS '02: Fully semantic
Q & A