Automatic Derivation of Meaningful Experiments for Hybrid Systems

Angelo E. M. Ciarlini, PUC-Rio, Brazil
Thom Frühwirth, LMU, Munich, Germany

DExVal Project → GMD/CNPq
German/Brazilian Cooperation Program
Software Validation and Testing

• “Are we building the right thing?”
• Reveal bugs → “good” input values
• Our approach:
  – Verification + Test Data Derivation
  – Hybrid Automata → Constraint Logic Programming (CLP)
  – Situations → expressive logic → constraints
  – Symbolic execution
  – Remaining constraints → input values
Hybrid Automata

- Variables
- States: name, invariant and iteration
- Transitions: source state, target state, guarded actions and events
- Concurrent timed hybrid automata
Constraint Logic Programming

- Logic Programming (LP): rules, search, backtracking
- Constraint Solving (CS): special-purpose algorithms
- Tight integration: deterministic (CS) and non-deterministic (LP) processes
- Eg.: Accumulated constraints $X+Y>5$ and $Y>0$. If $X$ is bound to 6 then CS detects failure
DExVal Architecture

- Behaviour specified by the user
- Constraint Solving
- Remaining Constraints
- Projection
- Search
- Test Data Generation
- Test data
- Path
- Automata
Specification of a Test Situation

• Scenario:
  – instances of classes of automata
  – parameters and synchronization

• User-specified conditions
  – $X: t \rightarrow$ variable $X$ at time $t$
  – Exist. and universal quantification
  – Modalities: “since”, “until”, “always in the past”, always in the future”, “sometime in the past” and “sometime in the future”
Symbolic Execution

• Representation of automata
  automaton_name(invariant-ST1,OLD_VARS,CONSTRAINTS)
  automaton_name(iteration-ST1,OLD_VARS,NEW_VARS,CONSTRAINTS)
  automaton_name(transition-ST1-ST2, OLD_VARS,NEW_VARS,CONSTRAINTS)

• Execution
  – Automata in parallel
  – Constraints sent to CS during search
  – Output: path and remaining constraints
  – Iterative deepening
  – Integration of constraint solvers and dynamic addition of constraints (eg ∀t X:t > 20) → Constraint Handling Rules (CHR)
Test Data Derivation Algorithm

• Projection of remaining constraints onto only one variable (repr. an input value) → domain

• Choose value within the domain and assign it to the variable

• Re-evaluate constraints

• Get values for the other variables
Test Data Derivation Features

- Compatibility with diff. criteria (e.g. “mutants” and “coverage of paths”)
- Deterministic process
- Expressive language
- Concurrent hybrid automata
Bathroom Boiler Scenario

- **HIGH** - high temperature (constant)
- **LOW** - low temperature (constant)
- **WT** - cold water temperature at the pump (constant)
- **T** - temperature (simulated by the automaton Boiler)
- **H** - heater (automatic control - automaton Boiler)
  - H=0 (off), H=1 (on)
- **P** - pump (automatic control - automaton Pump)
  - P=0 (off), P=1 (on)
- **MAX** - maximum water volume (constant)
- **V** - water volume (simulated by the automaton Boiler)
- **S** - shower (external variable)
  - S=0 (off), S=1 (on)
- **G** - granularity - controls the speed with which the water volume and temperature vary (constant)
Example

• Condition: water_volume:i=10.0\land temperature:i<100\land \forall T \ (heater:T=0)

• Good values for temperature:i?

• Output: 47.181, 73.59 and 99.999
Concluding Remarks

• Importance of CLP
  – Verification and derivation of properties
  – Generation of test cases

• Current work
  – Integration
  – Enhancement of our specialized CS
  – Use of CHR to solve problems with projection (eg. non-linear constraints)