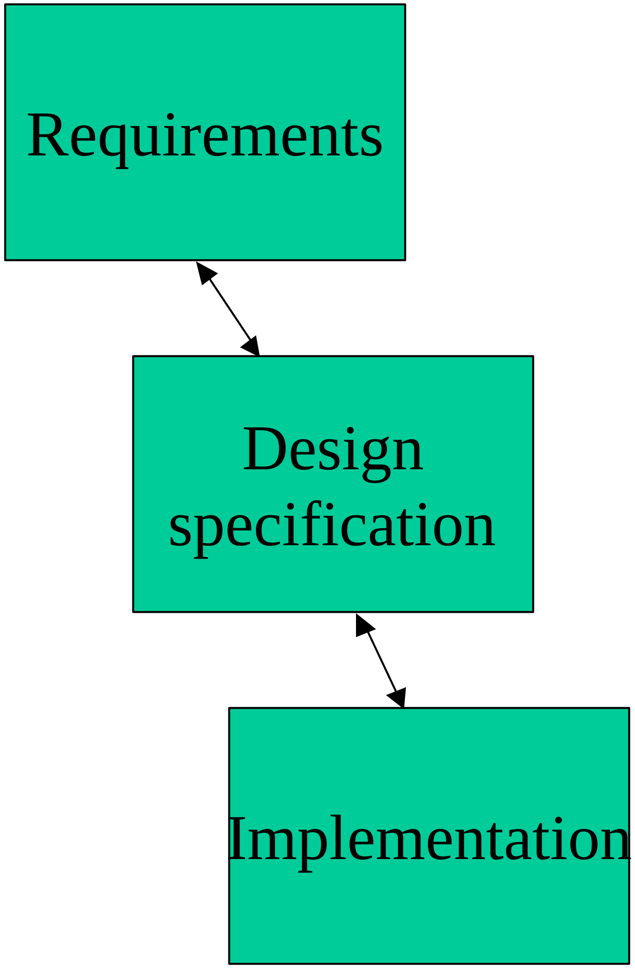
Temporal Logic Based Approach to Test Coverage and Generation

Hyoung S. Hong, Insup Lee, Oleg Sokolsky

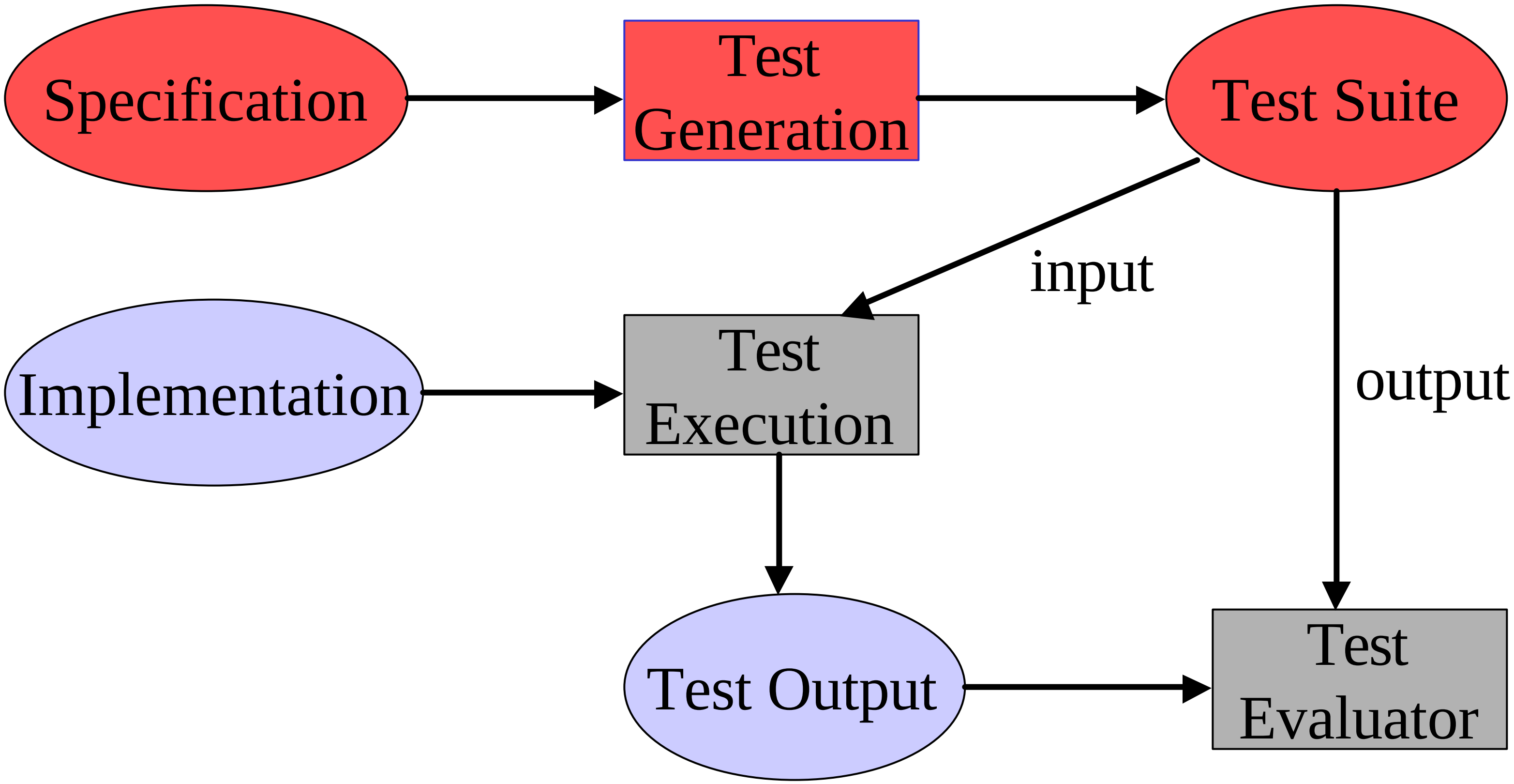
SDRL (Systems Design Research Lab)
RTG (Real-Time Systems Group)
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA

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

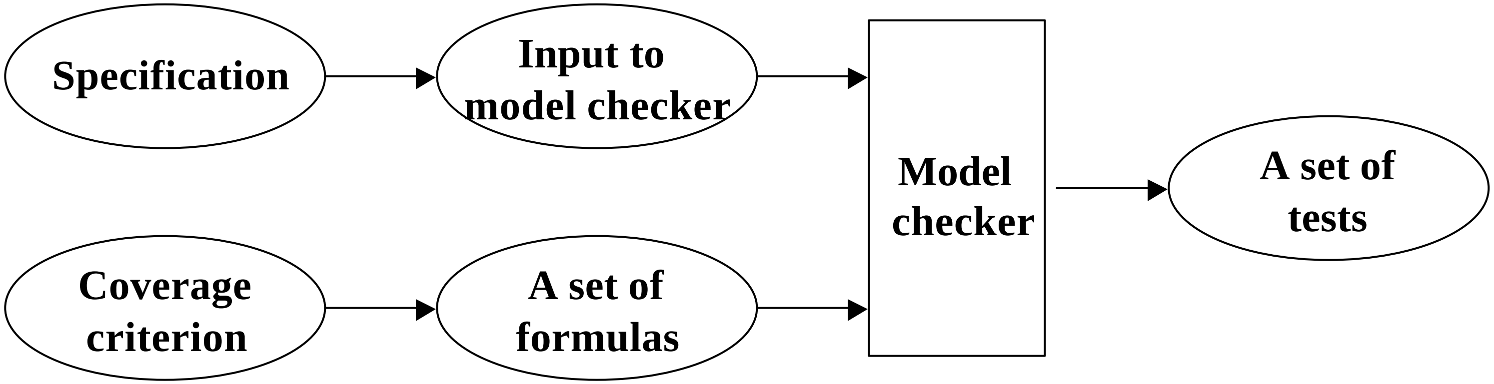


Specification-Based Testing

- 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

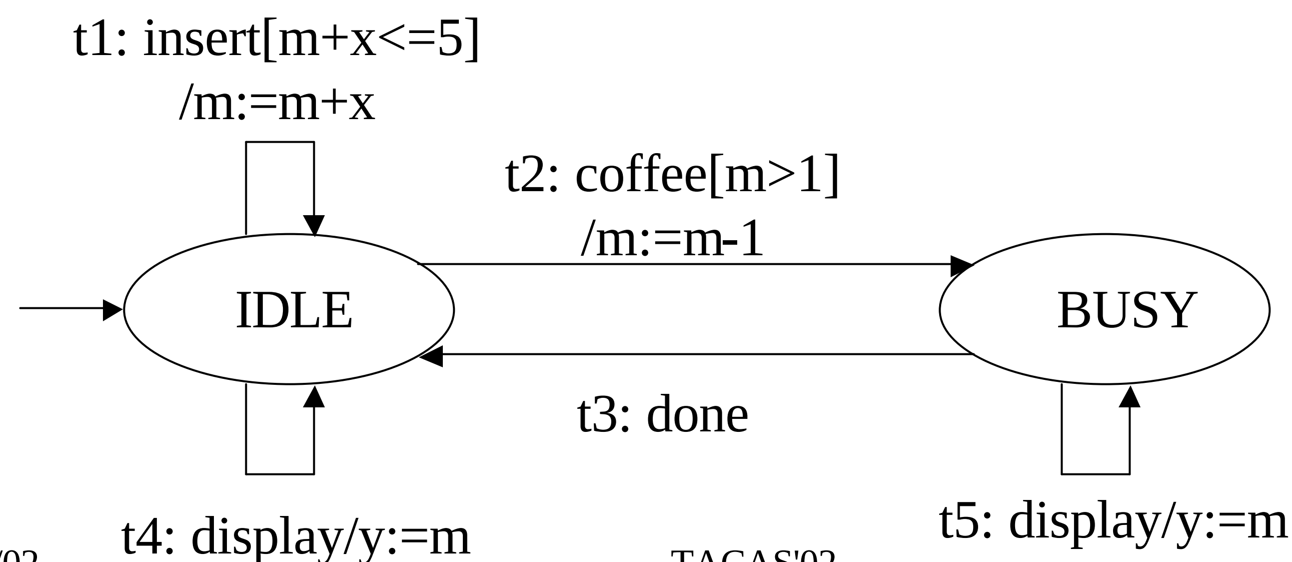
Our Approach

- 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



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

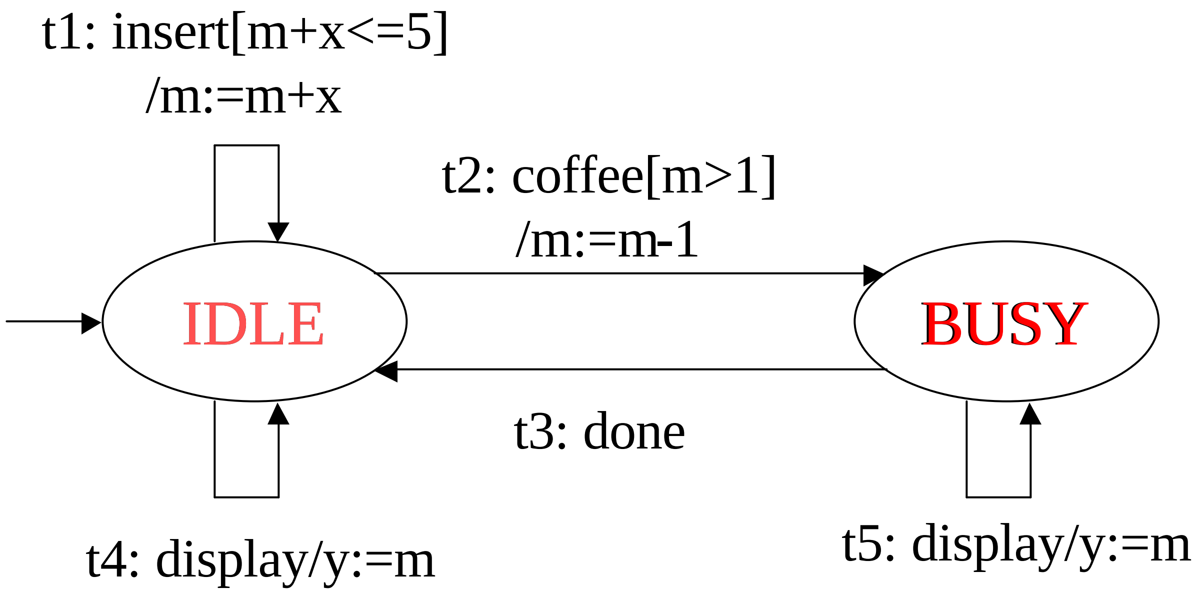


Coverage criteria in WCTL

- Each coverage criterion is represented by a set of temporal logic formulas
  - WCTL: a subset of CTL
    - Atomic propositions $p_1,\dots,p_n$
    - Temporal operators EX, EU, EF
    - Conjunctions have at most one non-atomic conjuncts
    - Negations can be applied only to atomic propositions
    - Unrestricted disjunctions
    - E.g.: $EF(p_1 \& EFp_2)$
  - WCTL formulas have linear witnesses

All-states coverage criterion

- Requires every state be covered at least once
- With every state *s*, associate EF(*s* & EF*exit*)

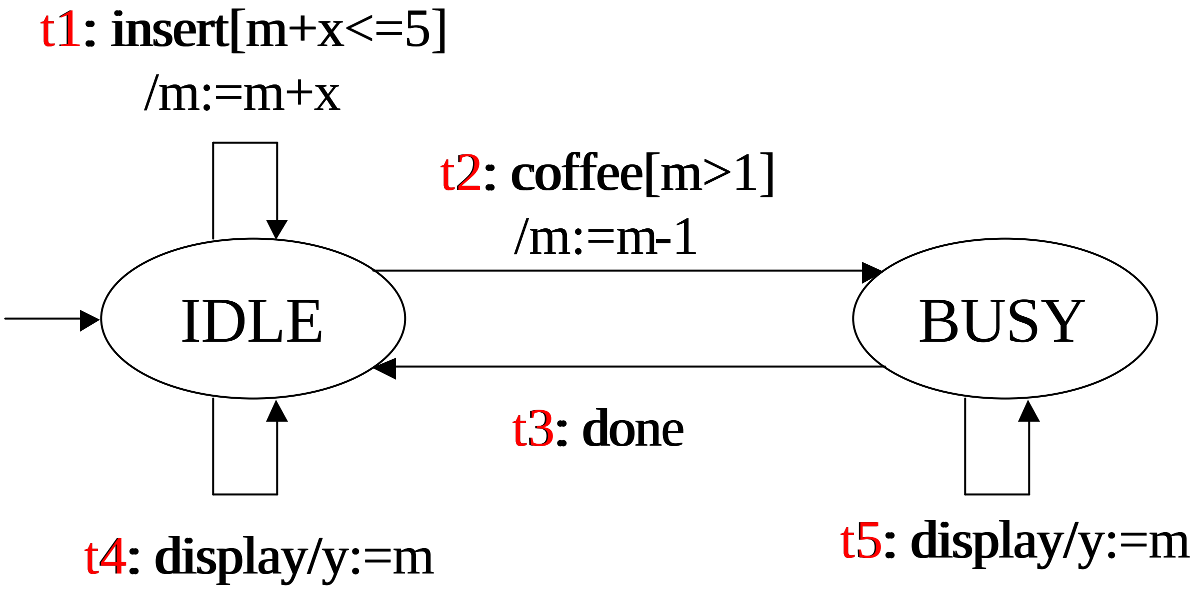


EF(*idle* & EF*exit*)

EF(*busy* & EF*exit*)

All-transitions coverage criterion

- Requires every transition be covered at least once
- With every transition *t*, associate EF(*t* & EF*exit*)



EF(*t1* & EF*exit*)
EF(*t2* & EF*exit*)
EF(*t3* & EF*exit*)
EF(*t4* & EF*exit*)
EF(*t5* & EF*exit*)

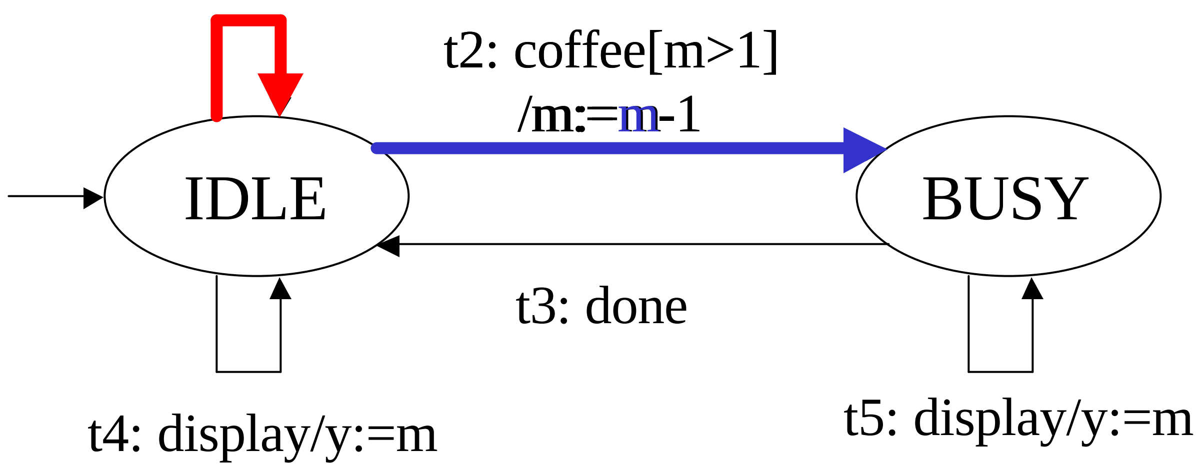
Data flow: definitions and uses

- 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

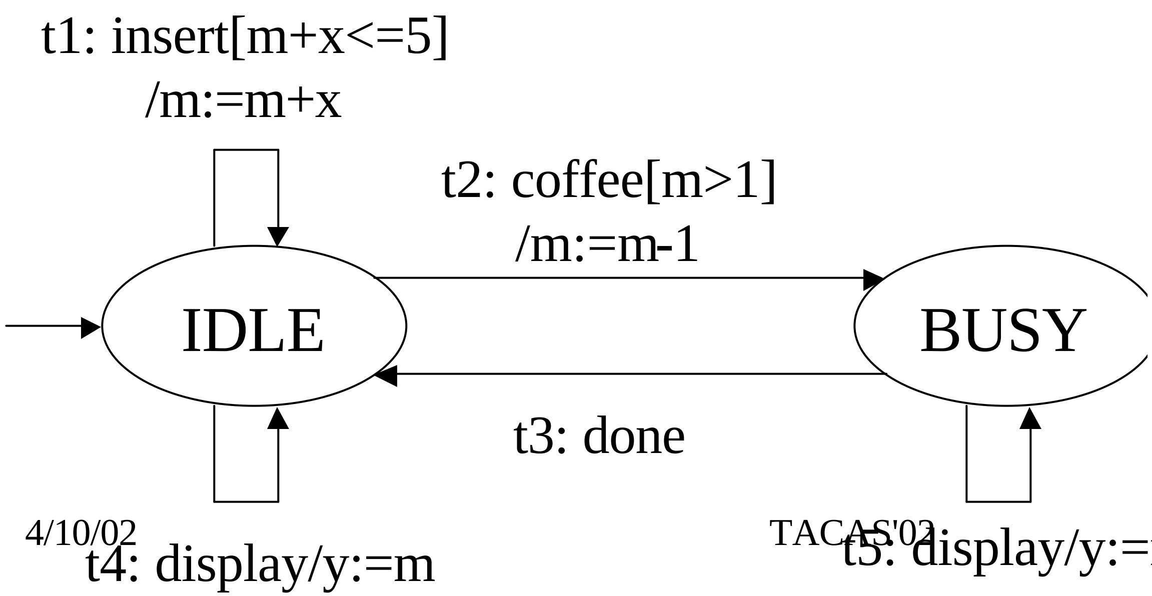
- With a definition-use pair (v, t, t'), associate
  - EF(t & EXE[!def(v) U (t' & EF*exit*)])
  - def(v) : disjunction of all transitions that define v

EF(*t1* & EXE[!(*t1* | *t2*) U (*t2* & EF*exit)]*)



Data-flow coverage criteria

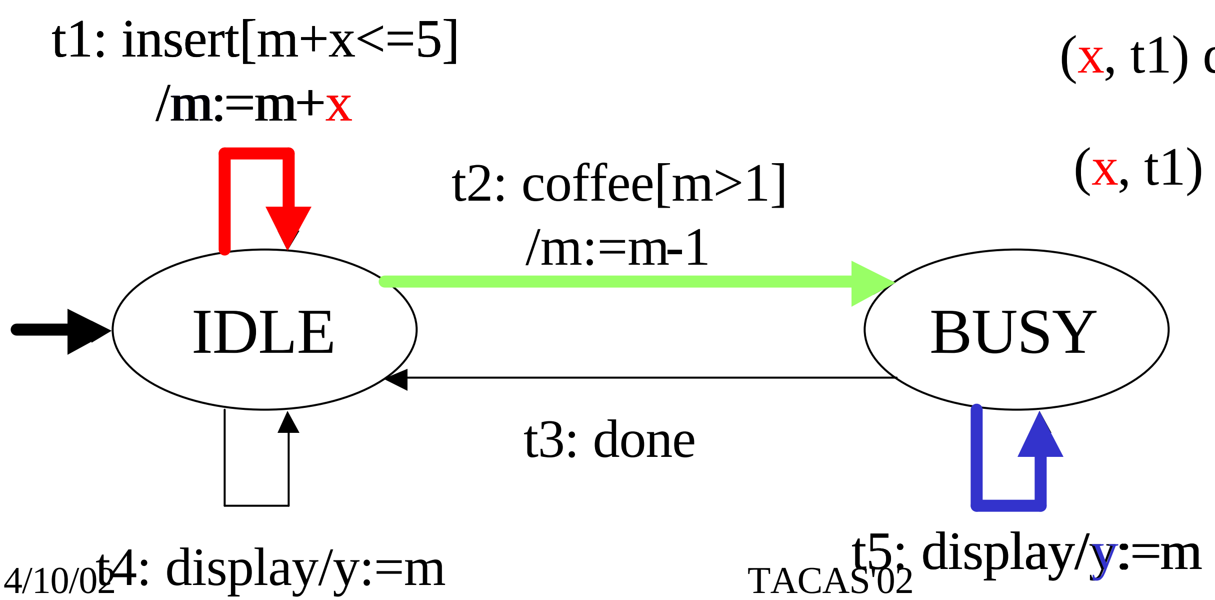
- *All-defs coverage criterion*
  - Requires a definition-clear path from **every** definition to **some** use be covered at least once
- *All-uses coverage criterion*
  - Requires a definition-clear path from **every** definition to **every** use be covered at least once



All-uses coverage criterion
EF(*t1* & EXE[!(*t1* | *t2*) U (*t1* & EF*exit*)])
EF(*t1* & EXE[!(*t1* | *t2*) U (*t2* & EF*exit*)])
EF(*t1* & EXE[!(*t1* | *t2*) U (*t4* & EF*exit*)])
EF(*t1* & EXE[!(*t1* | *t2*) U (*t5* & EF*exit*)])
EF(*t2* & EXE[!(*t1* | *t2*) U (*t1* & EF*exit*)])
EF(*t2* & EXE[!(*t1* | *t2*) U (*t2* & EF*exit*)])
EF(*t2* & EXE[!(*t1* | *t2*) U (*t4* & EF*exit*)])
EF(*t2* & EXE[!(*t1* | *t2*) U (*t5* & EF*exit*)])

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')



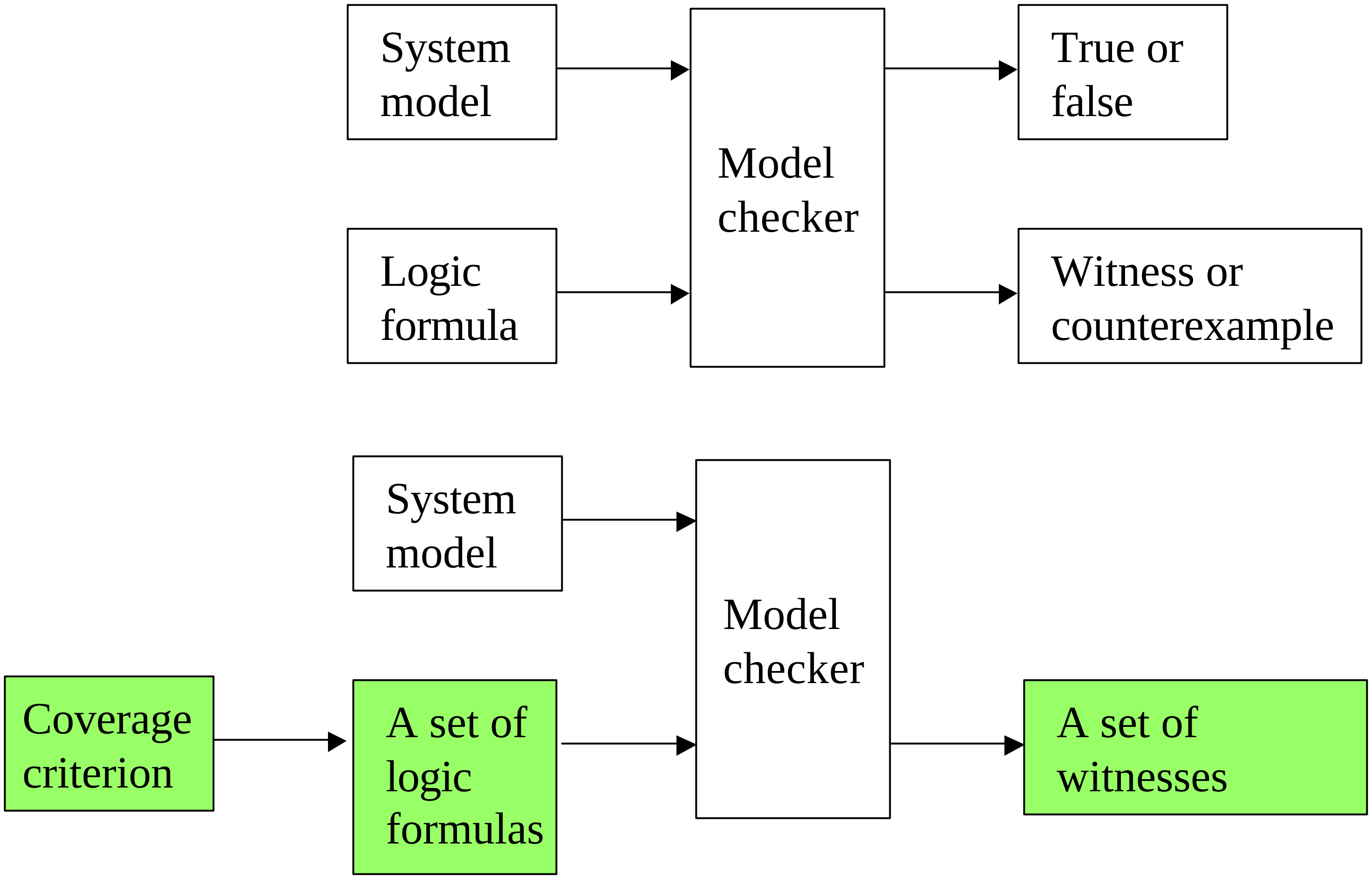
(x, t1) directly affects (m, t1)

(x, t1) affects (y, t5)

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

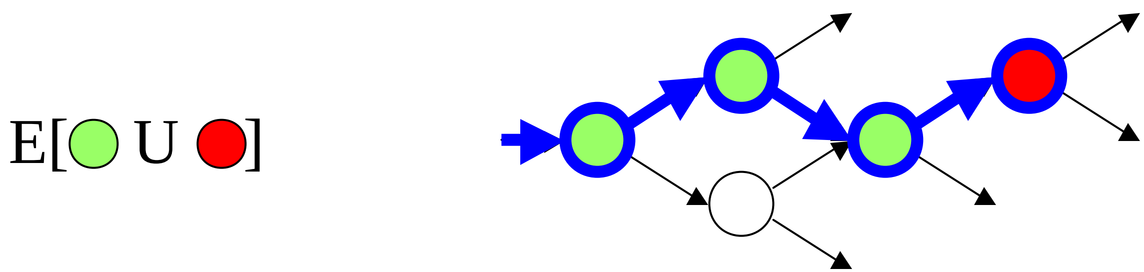
Test Generation



SDRL & RTG
University of Pennsylvania

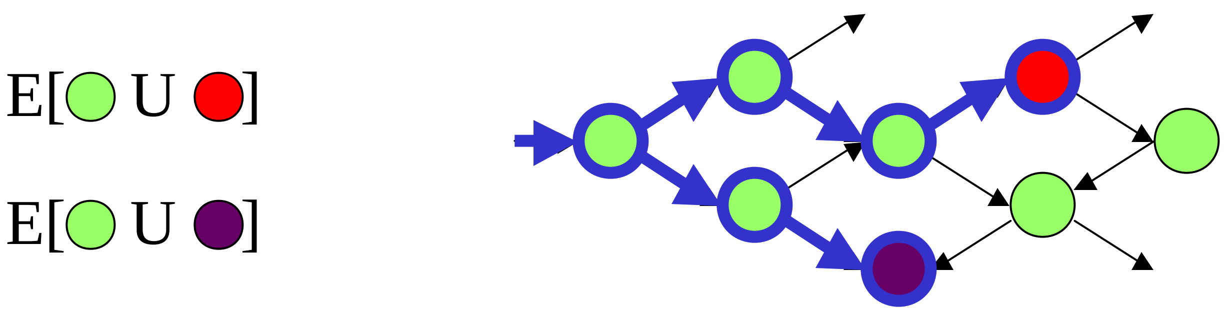
Witness Generation

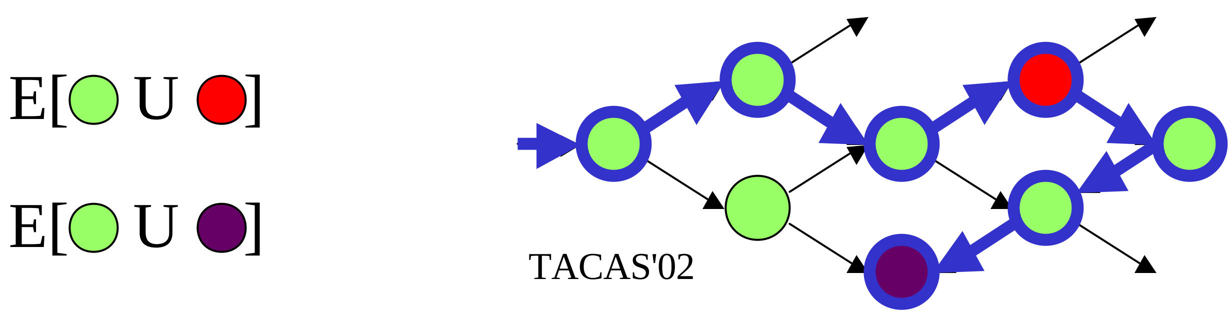
- Generating a witness for a formula
  - Cost: the length of a witness
  - A minimal-cost witness for a formula
    - Existing model checkers generate a minimal-cost witness by breadth-first search of state space



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)



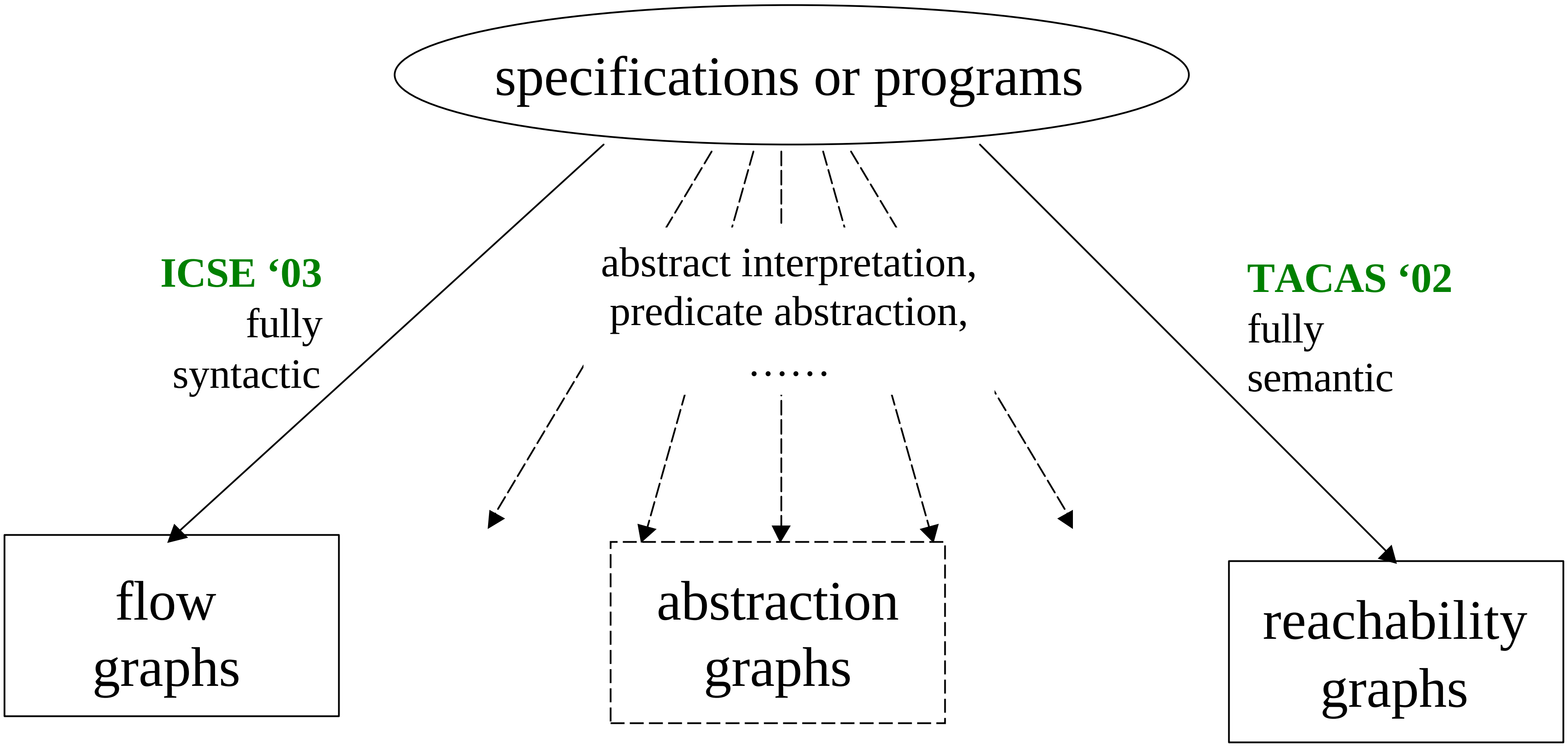


Model-based test generation

- 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



Q & A