χChek: A Multi-Valued Model-Checker

Marsha Chechik

Arie Gurfinkel

Benet Devereux

University of Toronto

Department of Computer Science

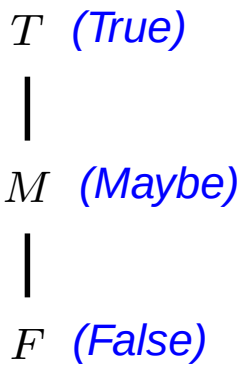
July 30, 2002

Multi-Valued Model-Checking

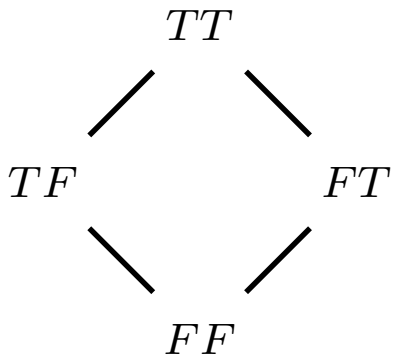
- Classical model-checking
  - logic: boolean logic
  - model: Kripke structure (state, transitions, variables)
  - property: CTL formula (eg. $AG(\texttt{snd} \rightarrow AF\texttt{ack})$)
  - result: the value of the formula on the model
    - either *true* or *false*, and a counterexample or a witness if possible
- Multi-valued model-checking
  - logic: a distributive lattice
    - meet ($\sqcap$) is conjunction, join ($\sqcup$) is disjunction, involution ($\neg$) is negation
  - model: χKripke structure
    - a Kripke structure with multi-valued variables and transitions
  - property: χCTL formula
    - CTL syntax, multi-valued semantics
  - result: the value of the formula on the model
    - a lattice value, and a counterexample or a witness if possible

Modeling with multi-valued logics

- Multi-valued logic
  - lattice elements are interpreted as truth values
  - $a \sqsubseteq b$ means "a is less true than b"
  - meet, join and involution are conjunction, disjunction and negation
- Partial, open or abstracted systems [BG CAV'99]
  - use 3-valued logic
  - T and F stand for *true* and *false*
  - M interpreted as: *maybe*, *unknown*, *undefined*



- Feature interaction, combining information from different sources [EC ICSE'01]
  - for two sources use 4-valued logic
  - TT and FF indicate agreement
  - TF and FT indicate disagreement



Multi-valued model-checking: theory

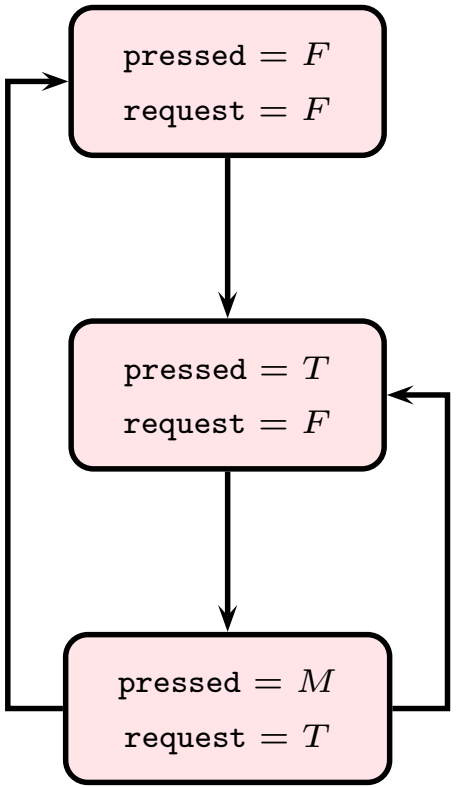
- Multi-valued logic
  - χKripke structures are used to provide semantics
  - the logic is represented explicitly
  - multi-valued model-checking is a unifying framework for:
    - reasoning about property equivalence and duality
    - defining simulation and refinement
    - discovering property preservation
    - reasoning in the presence of inconsistencies
- Query-checking [Chan CAV'99, BG LICS'01, GCD FSE'02]
  - query: a CTL formula with unknowns (eg. $AG(snd \rightarrow AF\ ?)$)
  - solution: the set of propositional formulas satisfying the query
  - as a multi-valued model-checking problem:
    - the lattice is interpreted as sets of propositional formulas
    - a query is represented by a χCTL formula
    - result of the model-checking is the solution to the query

Multi-valued model-checker: implementation

- Is multi-valued model-checking reducible to classical?. . . Yes!
- The difference is in decision diagrams used to represent the transition relation
  - reduction to classical — BDD vector
  - direct approach — Multi-Valued Decision Diagrams (MDD)
    - in MDD # of terminal nodes and branching factor equal the size of the logic
- BDD vector vs. MDD comparison
  - similar to BDD vector vs. ADD comparison for functions $\mathbb{B}^n \rightarrow \mathbb{N}$
  - depends on
    - the function being represented
    - variable ordering
    - operations performed, and more . . .
  - conclusion: MDD can perform better than BDD vector for some problems

Example

- A simple elevator controller
  - logic: 3-valued
  - number of floors: 3
  - a door that can be opened or closed
  - buttons inside the elevator and on floor landings
    - `pressed` — button's physical state
    - `request` — outstanding request



- Property:

  The elevator doors cannot stay closed forever.

  $AG(\neg \texttt{door} \rightarrow AF\ \texttt{door})$

Finally,

- Other features of χChek:
  - fairness
  - counterexample and witness generation
- χChek is available upon request from: xchek@cs.toronto.edu
- For papers on χChek and multi-valued model-checking see:
  `http://www.cs.toronto.edu/~chechik/publications.html`

χChek: Implementation

- Java based implementation
- Custom decision diagram library (Java)
- Decision Diagram:
  - custom Java-based implementation (BDD, ADD, MBTDD, MDD)
  - CUDD library (BDD, ADD)
- Supports counter-examples and witnesses
- Full support for model-checking with fairness