A Brief Introduction to Formal Methods

Moshe Y. Vardi, Rice U.
vardi@cs.rice.edu
Marta Kwiatkowska, U. Oxford
marta.kwiatkowska@cs.ox.ac.uk
**Formal Methods**: mathematically/logically-based techniques for the specification, development and verification of SW/HW systems.

**Key ideas:**
- Analysis/synthesis of models of systems
- *Precise* specification of intended behavior
Main Focus: Reactive System

- Reactive System: a system that reacts to external events
- Formal Model - Transducers

\[ T = (S, s_0, C, D, f, g) \]
- \( S \): finite state set, \( s_0 \) - initial state
- \( C \): input symbols, \( D \): output symbols
- \( f: S \times C \rightarrow S \) - transition function
- \( g: S \rightarrow D \) - output function

Trace: \((s_0, c_0, d_0), (s_1, c_1, d_1), \ldots\)
- \( s_{i+1} = f(s_i, c_i) \), \( d_i = g(s_i) \)
Nondeterminism

- **Nondeterminism**: modeling uncertainty
- Formal Model - Nondeterministic *Transducers*

\[ T=(S,s_0, C,D,f,g) \]
- \( S \): state set, \( s_0 \) - initial state
- \( C \): input symbols, \( D \): output symbols
- \( f: S \times C \to 2^S \) - *nondeterministic* transition fun.
- \( g: S \to D \) - output function

Trace: \( s_{i+1} \in f(s_i,c_i) \)
- Note: *probabilistic* treatment to come
Specification

- In particular: specify properties of traces

Examples: Linear Temporal Logic (LTL)
- always not (CS1 and CS2): mutual exclusion -- safety
- always (Request implies eventually Grant) - liveness

Psalm 34:14: “Depart from evil and do good”
**Analysis and Synthesis**

- **Analysis Problem**: Given a finite transducer \( T \) and an LTL formula \( \varphi \), do all traces of \( T \) satisfy \( \varphi \)? That is, is \( T \) a model of \( \varphi \)? (*Model Checking*)

- **Synthesis Problem**: Given an LTL formula \( \varphi \), is there a finite transducer \( T \) that is a model of \( \varphi \)? If so, construct one.
Analysis and Synthesis - Results

- **Analysis Problem**: Given a finite transducer $T$ and an LTL formula $\phi$, do all traces of $T$ satisfy $\phi$? That is, is $T$ a model of $\phi$? ([Model Checking])
  - Problem is PSPACE-complete.

- **Synthesis Problem**: Given an LTL formula $\phi$, is there a finite transducer $T$ that is a model of $\phi$? If so, construct one.
  - Problem is 2EXPTIME-complete.
Key Algorithmic Ideas - Automata

- **Models(φ) is a subset of \((2^{\text{Prop}})^\omega)\**
  - A formal language of infinite words
  - Can also be described by an automaton in infinite words
  - Algorithmic translation from logic to automata

- **Mature algorithmic theory for analysis**
  - Academic and industrial tools
Verification and Planning

- **Analysis Problem**: Given a finite transducer \( T \) and an LTL formula \( \phi \), do all traces of \( T \) satisfy \( \phi \)?

- **Answer**:
  - \( T \) is a model of \( \phi \) - MC says “Yes”
  - \( T \) is not a model of \( \phi \); MC says “No” and provides a counterexample trace

- **Bonus**: An algorithm to find system trace that satisfies a temporal property - *planning*!
  - Applications to robotics.
**Key Algorithmic Ideas - Games**

- **Synthesis**: Can be viewed as an infinite-round *game* between system and environment
  - Environments chooses input symbols.
  - System chooses output symbols.
  - System wants to satisfy $\phi$.
  - System wins if it has a *winning strategy*.
  - Strategy can be written as a transducer.

- **Elegant algorithmic theory for synthesis**
  - Academic tools
Beyond Finite Systems

- **Model Checking:** Is $L(T)$ contained in $L(\varphi)$?

- **Language Containment:** $L_1$ contained in $L_2$?
  - Decidable when both $L_1$ and $L_2$ are regular.
  - Undecidable if $L_2$ is context free.

- Beyond finite systems:
  - Model checking of recursive systems
  - Wrt “weak recursive” LTL (i.e., match calls and returns)

- Academic and industrial tools
Fundamental Challenge: Scalability

- Theoretical/practical computational complexity
  - Theoretical: worst case -- too pessimistic!
  - Practical: algorithmic behavior on real-world problem instances

- Good news: We can solve industrial problems of nontrivial size

- Bad news: It is hard to overcome exponential scaling - “climbing Mount Exponential”
  - “State-Explosion Problem”
Boolean Satisfiability (SAT); Given a Boolean expression, using “and” (&) “or”, (|) and “not” (!), is there a satisfying solution (an assignment of 0’s and 1’s to the variables that makes the expression equal 1)?

Example:

\((\neg x_1 \ | x_2 \ | x_3) \& (\neg x_2 \ | \neg x_3 \ | x_4) \& (x_1 \ | x_3 \ | x_4)\)

Solution: \(x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 1\)
Modern SAT Solving

**CDCL** = conflict-driven clause learning
- Backjumping
- Smart unit-clause preference
- Conflict-driven clause learning
- Smart choice heuristic
- Restarts

**Key Tools:** GRASP, 1996; Chaff, 2001

**Current capacity:** millions of variables
“Moore’s Law for SAT Solving”

Some Experience with SAT Solving

Sanjit A. Seshia

Speed-up of 2012 solver over other solvers

Solver

- Graep (2000)
- zChaff (2001)
- B ekM1 (2002-03)
- zChaff (2003-04)
- Slige (2004)
- Minisat + SatElite (2005)
- Rea+ SatElite (2007)
- Precon (2008)
- Cryptominisat (2010)
- Glucose 2.0 (2011)
- Glucose 2.1 (2012)
Formal Methods:
- Motivated by real-world problems
- Rich algorithmic theory
- Academic and industrial tools
- Impressive progress wrt real-world complexity
- Scalability challenge
- Current frontier: quantitative analysis