Symbolic Model Checking $10^{20}$
States and Beyond

J. R. Burch, E. M. Clarke,
K. L. McMillan, D. L. Dill
and L. J. Hwang
Goal of the Talk

The application of the BDD representation in the model checking problem
Acknowledgment

The following materials have been used in this talk:

- Course materials of CSC2108 in University of Toronto, by Prof. Marsha Chechik


- “Model Checking”, by Clarke, Grumberg, Peled, 1999, MIT Press
Outline

- Model Checking
  - System Modeling – Kripke Structure
  - Temporal Logic Specification – CTL

- CTL Model Checking
  - Explicit Labeling Algorithm
  - State Explosion Problem
  - Symbolic Model Checking

- Symbolic Model Checking
  - Mu-Calculus
  - Symbolic Model Checking with BDDs
What is Model Checking?

- **SW/HW artifact**
- **Model Extraction**
- **Model of System**
- **Checker Engine**
- **Correctness properties**
- **Translation**
- **Temporal logic**
- **Answer + Counter-example**

**Correctness properties**

**Temporal logic**

**Model Extraction**

**Checker Engine**

**SW/HW artifact**

**Model of System**
System Modeling – Kripke Structure

Conventional state machines

\[ M = \langle S, A, s_0, I, R \rangle \]

- \( S \) is a (finite) set of states
- \( A \) is a (finite) set of atomic propositional variables
- \( s_0 \) is a unique initial state (\( s_0 \in S \))
- \( I : S \rightarrow 2^A \) is a labelling function that maps each state to the set of propositional variables that hold in it
- \( R \subseteq S \times S \) is a (total) transition relation, that is, for every state \( s \in S \) there is a state \( s' \in S \) such that \( R(s, s') \)
Kripke Structure – Cont’d

- Model - a tree of computation paths
- Finite number of states, but infinite path
- Example:

![Kripke Structure](image)

Tree of computation
Temporal Logic

✓ Equipping the propositional logic with the notion of time
✓ The truth value varies over time

✓ Path Quantifiers
  - A – universal path quantifier (for all the paths....)
  - E – existential path quantifier (there exists a path .....)

✓ Temporal Quantifiers
  - X – next
  - F – future/eventually
  - G – global/always
  - U – until
CTL: Computation Tree Logic

- Branching time logic
- Time can branch into several streams
- No control over which branch is taken
- In CTL, no temporal operator can appear unless quantified
- Eight quantified temporal operators:
  - EX, EF, EG, EU, AX, AF, AG, AU
Examples

EX (exists next)

EG (exists global)

AX (all next)

AG (all global)
Examples - Cont’d

EF (exists future)

AF (all future)

EU (exists until)

AU (all until)
CTL – Cont’d

 principio

 **Syntax:**

 \[ \Phi ::= \text{true} \mid \text{false} \mid a \mid b \mid c \mid \ldots \]

 | \[ \neg \Phi \]
 | \[ \Phi_1 \lor \Phi_2 \]
 | \[ \text{EX} \Phi \]
 | \[ E (\Phi_1 \lor \Phi_2) \]

 **Others:**

 \[ \text{AX} \phi = \neg \text{EX} \neg \phi \]
 \[ \text{EF} \phi = E(\text{true} \lor \phi) \]
 \[ \text{AG} \phi = \neg \text{EF} \neg \phi \]
 \[ \text{EG} \phi = \neg \text{AG} \neg \phi \]
 \[ \text{AF} \phi = \neg \text{EG} \neg \phi \]
 \[ \text{A} (\phi_1 \lor \phi_2) = \neg (E [\neg \phi_2 \lor (\neg \phi_1 \land \neg \phi_2)] \lor \text{EG} \neg \phi_2) \]
Examples:

Properties that hold:

- $(\text{EX } p)(s_0)$
- $(\text{A}[p \lor q])(s_0)$

Properties that fail:

- $(\text{A}[\neg p \lor q])(s_0)$
- $(\text{EX } \text{AF } p)(s_0)$
CTL Model Checking

Given:
- Kripke structure $K$
- Temporal logic formula $\varphi$

Check:
- $\varphi$ holds in $K$?

Explicit Labeling Algorithm:
- Label states of $K$ with sub-formulas of $\varphi$ that are satisfied there and working outwards towards $\varphi$.
- Output states labeled with $\varphi$
Explicit Labeling Algorithm

**EX** $\phi$

- Label any state with **EX** $\phi$ if all of its successors are labeled with $\phi$

- Diagram:

**AF** $\phi$

- If any state $s$ is labeled with $\phi$, label it with **AF** $\phi$

- Repeat:
  - Label any state with **AF** $\phi$ if all of its successors are labeled with **AF** $\phi$
  - Until there is no change
Explicit Labeling - Example

Check AG EF y

\[ \text{AG EF } y = \neg \text{EF}(\neg \text{EF } y) \]
\[ S(y) = \{s2\} \]
\[ S(\text{EF } y) = \{s2, s1, s0\} \rightarrow S(\neg \text{EF } y) = \{\} \]
\[ S(\text{EF}(\neg \text{EF } y)) = \{\} \rightarrow S(\neg \text{EF}(\neg \text{EF } y)) = \{s0, s1, s2\} \]
Symbolic Model Checking

- **Explicit Labeling Algorithm**
  - Graph-based
  - Recursively go through the structure of the CTL property...
  - State explosion problem

- **Symbolic Modeling Checking**
  - Represents states symbolically (instead of listing the states)
  - Represents transition relations symbolically
  - Use some efficient data structures (e.g. BDD) to encode these
Symbolic Representation

- \( S_0 \rightarrow \neg x \land \neg y \)
- \( S_1 \rightarrow x \land \neg y \)
- \( S_2 \rightarrow x \land y \)

\( R = R(s_0, s_1) \lor R(s_1, s_0) \lor R(s_0, s_2) \lor R(s_2, s_2) \)
Model Checking using Sets of States

- Computing SAT(φ)
  - Give a CTL formula φ and a model, computes the set of states \( s \in S \) satisfying \( \phi \)
  - Check whether the initial states are included

- \( \phi \) is true : return \( S \)
- \( \phi \) is false : return \( \emptyset \)
- \( \phi \) is atomic : return \( \{ s \in S \mid \phi \in L(s) \} \)
- \( \phi \) is \( \neg \phi_1 \) : return \( S \setminus \text{SAT}(\phi_1) \)
- \( \phi \) is \( \phi_1 \land \phi_2 \) : return \( \text{SAT}(\phi_1) \cap \text{SAT}(\phi_2) \)
- \( \phi \) is \( \phi_1 \lor \phi_2 \) : return \( \text{SAT}(\phi_1) \cup \text{SAT}(\phi_2) \)
- ......
Monotone Function

Let $S$ be a set of states and $F: P(S) \rightarrow P(S)$ a function on the power set of $S$.

1. We say that $F$ is monotone if $X \subseteq Y$ implies $F(X) \subseteq F(Y)$ for all subsets $X$ and $Y$ of $S$.

2. A subset $X$ of $S$ is called a fixed point of $F$ if $F(X) = X$.

Monotone functions always have a least and a greatest fixed point.

The semantics of EG, AF and EU can be expressed via greatest and least fixed points of monotone functions on $P(S)$.
Fixpoint

 средством fixpoint

Greatest fixpoint

Def: Y = F(Y) \land \forall X \cdot X = F(X) \Rightarrow X \subseteq Y

Computed by: F^n(S) [by Knaster-Tarski Theorem]

Written as: \nu X. F(X)

Least fixpoint

Y = F(Y) \land \forall X \cdot X = F(X) \Rightarrow Y \subseteq X

Computed by: F^n(\emptyset) [by Knaster-Tarski Theorem]

Written as: \mu X. F(X)
Fixpoint Characteristics of CTL

Adequate set (EX, EG, EU)

1. $\text{SAT}_{EX}(\phi)$
   - $\{s_0 \in S \mid s_0 \rightarrow s_1 \text{ for some } s_1 \in \text{SAT}(\phi)\}$
   - Image computation

2. $\text{SAT}_{EG}(\phi)$
   - Intuition: greatest fixpoint: infinite # of iterations
   - $\text{SAT}(\text{EG } \phi) = \text{SAT}(\phi) \cap \text{SAT}(\text{EX EG } \phi)$
   - So, $\text{SAT}(\text{EG } \phi)$ is a fixpoint of
     $F(X) = \text{SAT}(\phi) \cap \text{SAT}(\text{EX } X)$, which is monotone
   - $\text{SAT}(\text{EG } \phi) = \nu X. F(X)$
3. \( \text{SAT}_{EU}(\varphi, \psi) \)

\( \triangleright \) Intuition: least fixpoint: finite # of iterations

\( \triangleright \) \( E[\varphi \ U \psi] = \psi \lor (\varphi \land \text{EXE}[\varphi \ U \psi]) \)

\( \triangleright \) \( \text{SAT}(E[\varphi \ U \psi]) = \text{SAT}(\psi) \cup (\text{SAT}(\varphi) \cap \text{SAT}(\text{EXE}[\varphi \ U \psi])) \)

\( \triangleright \) So, \( \text{SAT}(E[\varphi \ U \psi]) \) is a fixpoint of

\( F(X) = \text{SAT}(\psi) \cup (\text{SAT}(\varphi) \cap \text{SAT}(\text{EXE}(X))), \) which is monotone

\( \triangleright \) \( \text{SAT}(E[\varphi \ U \psi]) = \mu X. F(X) \)
Symbolic Model Checking with BDDs

- Construct BDD for the transition relation

- Compute a BDD representing all states that satisfying the formula – SAT

- Check if initial states are included
Symbolic Model Checking with BDDs

\[ MC(p) = \]

- \( p \in \text{atomic} \): return \( \text{BuildBDD}("p") \)
- \( p = \neg \varphi \): return \( \text{Apply}(\neg, MC(p)) \)
- \( p = \varphi_1 \land \varphi_2 \): return \( \text{Apply}(\land, MC(\varphi_1), MC(\varphi_2)) \)
- \( p = \varphi_1 \lor \varphi_2 \): return \( \text{Apply}(\lor, MC(\varphi_1), MC(\varphi_2)) \)
- \( p = \text{EX} \varphi \): return \( \text{existQuantify}(V', \text{Apply}(\land, R, \text{Prime}(MC(\varphi)))) \)
- \( p = E[\varphi U \psi] \): \( Q_0 = \text{BuildBDD}(\text{false}) \)
  \[
  Q_{i+1} = \text{Apply}(\lor, MC(\psi), \text{Apply}(\land, MC(\varphi), \text{MC(EX Q}_i)))
  \]
  return \( Q_n \) when \( Q_n = Q_{n+1} \)
- \( p = \text{EG} \varphi \): \( Q_0 = \text{BuildBDD}(\text{true}) \)
  \[
  Q_{i+1} = \text{Apply}(\land, MC(\varphi), MC(\text{EX Q}_i))
  \]
  return \( Q_n \) when \( Q_n = Q_{n+1} \)
Symbolic Model Checking - Example

Check $\text{EG } y$

$Q_0 = \text{true}$

$Q_1 = y \land \text{EX}(Q_0) = y$

$Q_2 = y \land \text{EX}(Q_1) = y \land ((x \land y) \lor (\neg x \land \neg y))$

$= (y \land x \land y) \lor (\neg x \land \neg y \land y)$

$= x \land y$

$Q_3 = y \land \text{EX}(Q_2) = y \land ((x \land y) \lor (\neg x \land \neg y))$

$= x \land y$

So, $\text{EG } y = x \land y \Rightarrow \{s2\}$
Mu-Calculus

Model: $M = (S, T, L)$, where

- $S$ – non-empty set of states
- $T$ – a set of transitions $T$, such that $\forall a \in T, a \subseteq S \times S$
- $L : S \rightarrow 2^{AP}$ that gives the set of atomic proposition true in a state

Operators

- $\neg$, $\land$, $\lor$
- $[a] - AX$
- $<a> - EX$
- $\nu Q.f$ and $\mu Q.f$ (greatest fixpoint and least fixpoint)
Mu-Calculus – Cont’d

Translating CTL formulas into Mu-Calculus: $\text{[Tr(\[ )]}$

- $\text{Tr}(p) = p$
- $\text{Tr}(\neg f) = \neg \text{Tr}(f)$
- $\text{Tr}(f \land g) = \text{Tr}(f) \land \text{Tr}(g)$
- $\text{Tr}(f \lor g) = \text{Tr}(f) \lor \text{Tr}(g)$
- $\text{Tr}(EX \ f) = <a>\text{Tr}(f)$
- $\text{Tr}(E[f U g]) = \mu Y.(\text{Tr}(g) \lor (\text{Tr}(f) \land <a>Y))$
- $\text{Tr}(EG \ f) = \nu Y.(\text{Tr}(f) \land <a>Y)$

Example: $\text{Tr}(EG(E[p U q])) =$

$\nu Y.(\mu Z.(q \lor (p \land <a>Z)) \land <a>Y)$
Summary

- The naïve approach of CTL model checking is in graph-based

- State explosion problem

- The complexity can be reduced by representing the states and the transitions in the model symbolically

- With the fixpoint characteristics, the CTL model checking problem into the form of the Mu-Calculus

- The new approach can take advantage of capturing the model with the BDD representation and doing model checking in the form of BDDs in the algorithm