Edmund M. Clarke
School of Computer Science
Carnegie Mellon University
- Try 4195835 – 4195835 / 3145727 * 3145727.
  In 94’ Pentium, it doesn’t return 0, but 256.
- Intel uses the SRT algorithm for floating point division. Five entries in the lookup table are missing.
- Cost: $400 - $500 million
- Xudong Zhao’s Thesis on Word Level Model Checking
Model checking is an automatic verification technique for finite state concurrent systems.

Developed independently by Clarke and Emerson and by Queille and Sifakis in early 1980’s.

Specifications are written in propositional temporal logic. (Pnueli 77)

Verification procedure is an intelligent exhaustive search of the state space of the design.
Advantages of Model Checking

- No proofs!!! (Algorithmic rather than Deductive)
- Fast (compared to other rigorous methods such as theorem proving)
- Diagnostic counterexamples
- No problem with partial specifications
- Logics can easily express many concurrency properties
Main Disadvantage

State Explosion Problem:

2-bit counter

n-bit counter has $2^n$ states
Main Disadvantage (Cont.)

- n states, m processes
- $n^m$ states
State Explosion Problem:

Unavoidable in worst case, but steady progress over the past 28 years using clever algorithms, data structures, and engineering
Determines Patterns on Infinite Traces

Atomic Propositions

Boolean Operations

Temporal operators

LTL - Linear Time Logic (Pn 77)

- $a$ \textit{“a is true now”}
- $X a$ \textit{“a is true in the neXt state”}
- $F a$ \textit{“a will be true in the Future”}
- $G a$ \textit{“a will be Globally true in the future”}
- $a \ U \ b$ \textit{“a will hold true Until b becomes true”}
Determines Patterns on Infinite Traces

Atomic Propositions

Boolean Operations

Temporal operators

- \(a\) “a is true now”
- \(X a\) “a is true in the next state”
- \(F a\) “a will be true in the future”
- \(G a\) “a will be globally true in the future”
- \(a U b\) “a will hold true until b becomes true”
Determines Patterns on Infinite Traces

Atomic Propositions
Boolean Operations
Temporal operators

- **a**: “a is true now”
- **X a**: “a is true in the next state”
- **Fa**: “a will be true in the Future”
- **Ga**: “a will be Globally true in the future”
- **a U b**: “a will hold true Until b becomes true”
Determines Patterns on Infinite Traces

Atomic Propositions
Boolean Operations
Temporal operators

- **a**
  - "a is true now"
- **X a**
  - "a is true in the next state"
- **Fa**
  - "a will be true in the Future"
- **Ga**
  - "a will be Globally true in the future"
- **a U b**
  - "a will hold true Until b becomes true"
Determines Patterns on Infinite Traces

Atomic Propositions
Boolean Operations
Temporal operators

\( a \) \quad "a is true now"
\( Xa \) \quad "a is true in the next state"
\( Fa \) \quad "a will be true in the Future"
\( Ga \) \quad "a will be Globally true in the future"
\( a \cup b \) \quad "a will hold true Until b becomes true"
Branching Time (EC 80, BMP 81)
CTL: Computation Tree Logic

EF g
“g will possibly become true”
AF g  "g will necessarily become true"
CTL: Computation Tree Logic

AG $g$  "g is an invariant"
CTL: Computation Tree Logic

EG $g$  “$g$ is a potential invariant”
CTL (CES83-86) uses the temporal operators

\[ \text{AX, AG, AF, AU} \]
\[ \text{EX, EG, EF, EU} \]

CTL* allows complex nestings such as

\[ \text{AXX, AGX, EXF, ...} \]
Model Checking Problem

- Let $M$ be a state-transition graph.
- Let $f$ be the specification in temporal logic.
- Find all states $s$ of $M$ such that $M, s \models f$.

- CTL Model Checking: CE 81; CES 83/86; QS 81/82.
- LTL Model Checking: LP 85.
- Automata Theoretic LTL Model Checking: VW 86.
- CTL* Model Checking: EL 85.
Microwave Oven

State-transition graph describes system evolving over time.
The oven doesn’t **heat up** until the **door is closed**.

**Not** **heat_up** holds **until** **door_closed**

(\(\neg \text{heat}_\text{up}\) **U** **door_closed**
Model Checking

Hardware Description
(VERILOG, VHDL, SMV)

Informal Specification

Transition System
(Automaton, Kripke structure)

Temporal Logic Formula
(CTL, LTL, etc.)

compilation

algorithmic verification

manual
Counterexamples

Program or circuit

Transition System

Informal Specification

Temporal Logic Formula (CTL, LTL, etc.)

Safety Property:
bad state unreachable:
satisfied
Counterexamples

Program or circuit

Transition System

Informal Specification

Temporal Logic Formula
(CTL, LTL, etc.)

Safety Property:
bad state unreachable

Counterexample
Counterexamples

- Program or circuit
- Transition System
- Informal Specification
- Temporal Logic Formula (CTL, LTL, etc.)

Safety Property: bad state unreachable

Counterexample

Initial State
Hardware Example: IEEE Futurebus+

- In 1992 we used Model Checking to verify the **IEEE Future+ cache coherence protocol**.

- Found a number of **previously undetected errors** in the design.

- First time that a formal verification tool was used to find errors in an **IEEE standard**.

- Development of the protocol began in **1988**, but previous attempts to validate it were informal.
Four Big Breakthroughs on State Space Explosion Problem!

- **Symbolic Model Checking**
  Burch, Clarke, McMillan, Dill, and Hwang 90;
  Ken McMillan’s thesis 92

- **The Partial Order Reduction**
  Valmari 90
  Godefroid 90
  Peled 94
  (Gerard Holzmann’s SPIN)
Four Big Breakthroughs on State Space Explosion Problem!

- **Symbolic Model Checking**
  Burch, Clarke, McMillan, Dill, and Hwang 90;
  Ken McMillan’s thesis 92

  $10^{20}$ states

- **The Partial Order Reduction**
  Valmari 90
  Godefroid 90
  Peled 94
  (Gerard Holzmann’s SPIN)
Four Big Breakthroughs on State Space Explosion Problem!

- **Symbolic Model Checking**
  Burch, Clarke, McMillan, Dill, and Hwang 90;
  Ken McMillan’s thesis 92

  \(10^{100}\) states

- **The Partial Order Reduction**
  Valmari 90
  Godefroid 90
  Peled 94
  (Gerard Holzmann’s SPIN)
Four Big Breakthroughs on State Space Explosion Problem!

- **Symbolic Model Checking**
  Burch, Clarke, McMillan, Dill, and Hwang 90;
  Ken McMillan’s thesis 92

  $10^{120}$ states

- **The Partial Order Reduction**
  Valmari 90
  Godefroid 90
  Peled 94
  (Gerard Holzmann’s SPIN)
Four Big Breakthroughs on State Space Explosion Problem (Cont.)

- **Bounded Model Checking**
  - Biere, Cimatti, Clarke, Zhu 99
  - Using Fast SAT solvers
  - Can handle thousands of state elements

Can the given property fail in \(k\)-steps?

\[ I(V_0) \land T(V_0, V_1) \land \ldots \land T(V_{k-1}, V_k) \land \neg P(V_0) \lor \ldots \lor \neg P(V_k) \]

Initial state \(\rightarrow\) \(k\)-steps \(\rightarrow\) Property fails in some step

BMC in practice: Circuit with 9510 latches, 9499 inputs
BMC formula has \(4 \times 10^6\) variables, \(1.2 \times 10^7\) clauses
Shortest bug of length 37 found in 69 seconds
Four Big Breakthroughs on State Space Explosion Problem (Cont.)

- **Localization Reduction**
  - Bob Kurshan 1994

- **Counterexample Guided Abstraction Refinement (CEGAR)**
  - Clarke, Grumberg, Jha, Lu, Veith 2000
  - Used in most software model checkers
Given an abstraction function \( \alpha : S \rightarrow S_\alpha \), the concrete states are grouped and mapped into abstract states:

\[
\begin{align*}
&\text{M} \\
&\downarrow \quad \alpha \\
&\text{M}_\alpha
\end{align*}
\]
Theorem (Clarke, Grumberg, Long) If property holds on abstract model, it holds on concrete model.

Technical conditions:
- Property is universal i.e., no existential quantifiers
- Atomic formulas respect abstraction mapping

Converse implication is not true!
Spurious Behavior

AGAF red
“Every path necessarily leads back to red.”

Spurious Counterexample:
<go><go><go><go> ...
Artifact of the abstraction!
Automatic Abstraction

$M_\alpha$

Initial Abstraction

Refinement

Refinement

Correct!

Original Model

$M$

Spurious counterexample

Validation or Counterexample
CEGAR
CounterExample-Guided Abstraction Refinement

- Circuit or Program
- Initial Abstraction
- Abstract Model
- Refinement
- Model Checker
- Verification
- No error or bug found
  - Property holds
- Bug found
  - Simulation successful
- Counterexample
- Spurious counterexample
Future Challenge
Is it possible to model check software?

According to *Wired News* on Nov 10, 2005:

“When Bill Gates announced that the technology was under development at the 2002 Windows Engineering Conference, he called it the holy grail of computer science”
What Makes Software Model Checking Different?

- Large/unbounded base types: int, float, string
- User-defined types/classes
- Pointers/aliasing + unbounded #’s of heap-allocated cells
- Procedure calls/recursion/calls through pointers/dynamic method lookup/overloading
- Concurrency + unbounded #’s of threads
What Makes Software Model Checking Different?

- Templates/generics/include files
- Interrupts/exceptions/callbacks
- Use of secondary storage: files, databases
- Absent source code for: libraries, system calls, mobile code
- Esoteric features: continuations, self-modifying code
- Size (e.g., MS Word = 1.4 MLOC)
What Does It Mean to Model Check Software?

Combine static analysis and model checking

Use static analysis to extract a model $K$ from an abstraction of the program.

Then check that $f$ is true in $K$ ($K \models f$), where $f$ is the specification of the program.

- SLAM (Microsoft)
- Bandera (Kansas State)
- MAGIC, SATABS (CMU)
- BLAST (Berkeley)
- F-Soft (NEC)
Also according to *Wired News*:

“Microsoft has developed a tool called Static Device Verifier or SDV, that uses ‘Model Checking’ to analyze the source code for Windows drivers and see if the code that the programmer wrote matches a mathematical model of what a Windows device driver should do. If the driver doesn’t match the model, the SDV warns that the driver might contain a bug.”

(Ball and Rajamani, Microsoft)
Future Challenge
Can We Debug This Circuit?
“The p53 pathway has been shown to mediate cellular stress responses; p53 can initiate DNA repair, cell-cycle arrest, senescence and, importantly, apoptosis. These responses have been implicated in an individual's ability to suppress tumor formation and to respond to many types of cancer therapy.”


The protein p53 has been described as the guardian of the genome referring to its role in preventing genome mutation.

In 1993, p53 was voted molecule of the year by Science Magazine.
Questions?