Introduction to Model Checking

Angelo Montanari

Dipartimento di Scienze Matematiche,
Informatiche e Fisiche
Università di Udine, Italy

Formal verification and model checking

Unlike simulation and testing techniques, formal verification conducts an EXHAUSTIVE EXPLORATION of all possible behaviors of a system

The use of theorem proves, term rewriting systems, and proof checkers in formal verification: they are time consuming and they often require a great deal of manual intervention

(Temporal logic) MODEL CHECKING: an alternative verification technique developed independently by Clarke and Emerson in USA and by Queille and Sifakis in France at the beginning of the '80s

SPECIFICATIONS are expressed as formulae in a propositional temporal logic to be checked against STATE-TRANSITION SYSTEMS (is the transition system a model of the specification?) by using efficient search procedures

The distinctive features of model checking

The method of MODEL CHECKING: a desired property of the system is verified over a given system (the model) through an exhaustive (explicit or implicit) enumeration of all the states reachable by the system and the behaviors that traverse through them

Distinctive features of model checking:

- it is FULLY AUTOMATIC
- when the design fails to satisfy a given property, a COUNTEREXAMPLE is produced that exhibits a behavior which falsifies the property (useful insight to understand the reason for the failure and clues for fixing the problem)

Model Checking

In its original form, model checking is a simple (and very successful) method to decide the relation

$$\mathcal{M}, s \models \phi$$

where $\mathcal{M}$ is a temporal structure / Kripke structure (set of states plus accessibility relation), s is a designated state in $\mathcal{M}$, and ϕ is a formula of some temporal logic

Format of Kripke structures (over $P_1, \ldots, P_n$)

$$\mathcal{M} = (S, R, L)$$

where S is the state set, $R \subseteq S \times S$ is the transition relation, and $L : S \to 2^{\{P_1,\ldots,P_n\}}$ is the labeling function

Linear Time Logic (LTL)

Constructed from

- atomic propositions $P_1, \ldots, P_n$
- propositional connectives $\neg, \wedge, \vee, \rightarrow$, and $\leftrightarrow$
- unary temporal operators $X\cdot$ (next time), $F\cdot$ (eventually), $G\cdot$ (always), and the binary temporal operator $\cdot\mathcal{U}\cdot$ (until)

example of LTL formula:

$$G(P_1 \rightarrow X(\neg P_2 \mathcal{U} P_1))$$

"after any time where P_1 holds, P_2 is false until P_1 holds again"

$$\mathcal{M}, s \models \phi \Leftrightarrow \textbf{each path } \Pi = ss_1s_2\ldots \textbf{ through } \mathcal{M} \textbf{ satisfies } \phi$$
$$(\mathcal{M} \textbf{ satisfies } A\phi \textbf{ for short})$$

Computation Tree Logic (CTL)

Constructed from

- atomic propositions $P_1, \ldots, P_n$
- propositional connectives $\neg, \wedge, \vee, \rightarrow$, and $\leftrightarrow$
- unary temporal operators $EX\cdot$, $EF\cdot$, $EG\cdot$, $AX\cdot$, $AF\cdot$, $AG\cdot$, and the binary temporal operators $E(\cdot \mathcal{U} \cdot)$ and $A(\cdot \mathcal{U} \cdot)$

example of CTL formula:

$$AG(P_1 \rightarrow AX\ EFP_2)$$

"in each tree rooted at a son of a node where P_1 holds, there exists a path such that there exists a node in the path where P_2 holds"

$$\mathcal{M}, s \models \phi \Leftrightarrow \textbf{the tree unraveling of } \mathcal{M} \textbf{ at state } s \textbf{ satisfies } \phi$$

Model Checking I: the Case of LTL

(Manna, Pnueli) Given the behavior graph for $\mathcal{M}$ and the tableau $\mathcal{T}_{\neg\phi}$ for $\neg\phi$, look for initial (starting from s), fulfilling (satisfying all promises), and fair (satisfying fairness conditions) trails in $\mathcal{M} \times \mathcal{T}_{\neg\phi}$.

(Vardi, Wolper) Given $(\mathcal{M}, s)$ and the LTL formula ϕ, construct a Büchi automaton $\mathcal{A}$ which accepts those paths $\Pi = ss_1s_2\ldots$ through $\mathcal{M}$ which violates ϕ

$$\mathcal{A} = \mathcal{A}_{(\mathcal{M},s)} \times \mathcal{A}_{\neg\phi}$$

Check for nonemptiness of $\mathcal{A}$ ($L(\mathcal{A})$ is the set of bug scenarios)

Complexity: PSPACE-complete

linear in $|\mathcal{M}|$ $(= |S| + |R|)$
exponential in $|\phi|$

Model Checking II: the Case of CTL

(Clarke, Emerson) Given $\mathcal{M}$ and the CTL formula ϕ, do the following for the subformulas ψ of ϕ (according to an increasing order of their complexities):

$$\text{compute } [\psi] = \{r \in S | \mathcal{M}, r \models \psi\}$$

Finally, Check whether $s \in [\phi]$

Remark. If $\mathcal{M}$ is finite, this can be done effectively by fixed-point computations.

Complexity: Polynomial

linear in $|\mathcal{M}|$ $(= |S| + |R|)$
linear in $|\phi|$

Practical Applications

$\mathcal{M}$: representation of circuits, domains, protocols, ...

ϕ: formulation of desired properties

Task: Find states (or paths) in $\mathcal{M}$ which violate ϕ (bugs, error scenarios)

Use model checking as a debugging method

Problem: state space explosion

Possible ways out: symbolic model checking, partial order reduction, bounded model checking, k-liveness, IC3

Prominent tools: SMV (McMillan), NuSMV, nuXmv, SPIN (Holzmann), UPPAAL (Larsen)

Conferences: CAV, TACAS, and many others