

The Birth of Model Checking^{*}

Edmund M. Clarke

Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
`emc@cs.cmu.edu`

“When the time is ripe for certain things, these things appear in different places in the manner of violets coming to light in early spring.” (Wolfgang Bolyai to his son Johann in urging him to claim the invention of non-Euclidean geometry without delay [Vit88]).

1 Model Checking

Model Checking did not arise in a historical vacuum. There was an important problem that needed to be solved, namely Concurrent Program Verification. Concurrency errors are particularly difficult to find by program testing, since they are often hard to reproduce. Most of the formal research on this topic involved constructing proofs by hand using a Floyd-Hoare style logic. Probably, the best known formal system was the one proposed by Owicki and Gries [OG76] for reasoning about Conditional Critical Regions. Although I had written my thesis on the meta-theory of Hoare Logic [Cla77a, Cla77b, Cla78, Cla79a, Cla79c, Cla80] and was very familiar with the Owick-Gries proof methodology, I was quite skeptical about the scalability of hand constructed proofs. There had been some practical research on state exploration methods for communication protocols by Gregor Bochmann and others, but it was largely ignored by the “Formal Verification Community”. Also, in the late 1970’s, Pnueli [Pnu77] and Owicki and Lamport [OL82] had proposed the use of Temporal Logic for specifying concurrent programs. Although they still advocated hand constructed proofs, their work demonstrated convincingly that Temporal Logic was ideal for expressing concepts like mutual exclusion, absence of deadlock, and absence of starvation.

Allen Emerson and I combined the state-exploration approach with Temporal Logic in an efficient manner and showed that the result could be used to solve non-trivial problems. Here is a quote from our original 1981 paper [CE81]:

^{*} This research was sponsored by the National Science Foundation under grant nos. CNS- 0411152, CCF-0429120, CCR-0121547, and CCR-0098072, the US Army Research Office under grant no. DAAD19-01-1-0485, and the Office of Naval Research under grant no. N00014-01-1-0796. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

“The task of proof construction is in general quite tedious and a good deal of ingenuity may be required to organize the proof in a manageable fashion. We argue that proof construction is unnecessary in the case of finite state concurrent systems and can be replaced by a model-theoretic approach which will mechanically determine if the system meets a specification expressed in propositional temporal logic. The global state graph of the concurrent systems can be viewed as a finite Kripke structure and an efficient algorithm can be given to determine whether a structure is a model of a particular formula (i.e. to determine if the program meets its specification).”

1.1 What Is Model Checking?

The Model Checking problem is easy to state:

Let M be a Kripke structure (i.e., state-transition graph). Let f be a formula of temporal logic (i.e., the specification). Find all states s of M such that $M, s \models f$.

We used the term *Model Checking* because we wanted to determine if the temporal formula f was true in the Kripke structure M , i.e., whether the structure M was a *model* for the formula f . Some people believe erroneously that the use of the term “model” refers to the dictionary meaning of this word (e.g., a miniature representation of something or a pattern of something to be made) and indicates that we are dealing with an abstraction of the actual system under study.

Emerson and I gave a polynomial algorithm for solving the Model Checking Problem for the logic CTL. The figure below shows the structure of a typical Model Checking system. A preprocessor extracts a state transition graph from a program or circuit. The Model Checking engine takes the state transition graph and a temporal formula and determines whether the formula is true or not (Figure 1).

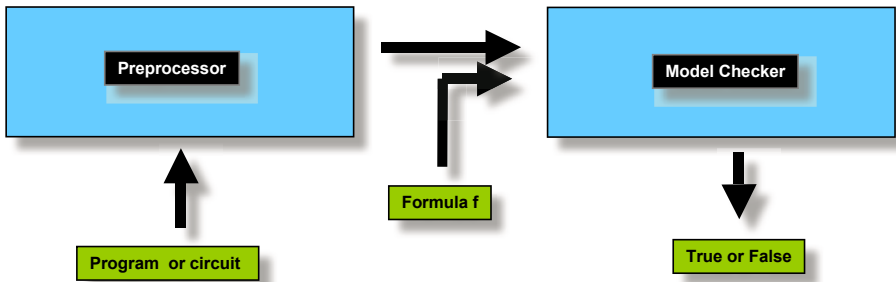


Fig. 1. Model Checker Structure

1.2 Advantages of Model Checking

Model Checking has a number of advantages compared to other verification techniques such as automated theorem proving or proof checking. A partial list of some of these advantages is given below: