Model Checking – My 27-year Quest to Overcome the State Explosion Problem

Ed Clarke
Computer Science Department, Carnegie Mellon University
Pittsburgh, PA

Turing Award 2007

Abstract

Model Checking is an automatic verification technique for state-transition systems that are finite-state or that have finite-state abstractions. In the early 1980's in a series of joint papers with my graduate students E.A. Emerson and A.P. Sistla, we proposed that Model Checking could be used for verifying concurrent systems and gave algorithms for this purpose. At roughly the same time, Joseph Sifakis and his student J.P. Queille at the University of Grenoble independently developed a similar technique. Model Checking has been used successfully to reason about computer hardware and communication protocols and is beginning to be used for verifying computer software. Specifications are written in temporal logic, which is particularly valuable for expressing concurrency properties. An intelligent, exhaustive search is used to determine if the specification is true or not. If the specification is not true, the Model Checker will produce a counterexample execution trace that shows why the specification does not hold. This feature is extremely useful for finding obscure errors in complex systems. The main disadvantage of Model Checking is the state-explosion problem, which can occur if the system under verification has many processes or complex data structures. Although the state-explosion problem is inevitable in worst case, over the past 27 years considerable progress has been made on the problem for certain classes of state-transition systems that occur often in practice. In this talk, I will describe what Model Checking is, how it works, and the main techniques that have been developed for combating the state explosion problem.

E. Denney, D. Giannakopoulou, C.S. Păsăreanu (eds.); The First NASA Formal Methods Symposium, pp. 1-1