Ed Clarke
FORE Systems Professor of Computer Science
Department of Computer Science
School of Computer Science

August 30, 2007 (11:00am)

Research Topic
Software Engineering

Research Problem
How can we practically verify specifications?

Problem Statement
Given the current sate of theory about formal models and the model checking systems in use, develop a new implementation of model checking techniques that can be practically and systematically applied to new specifications.

Operational Definitions
Formal model: A formal abstraction of anything that uses mathematical and logical terminology to provide a definition.
Model checking: A method of verifying that a model is correct, often by analyzing logical statements derived from the model.

Problem Description
Model checking is used to verify systems by an exhaustive search of the state space of the design. One of the advantages of model checking is that it can easily prove certain things do or do not happen; a good model can be used to verify things as general as the fact that a microwave oven will not start heating things until the door is closed. Moreover, if our property does not hold, it can provide states where it is not true and how to get to them. Use of formal models to verify software has not been as extensively solved as model checking due to some inherent issues with adapting the technique to analyze source code; these include the large size and complexity of programs and the large, potentially infinite, array of possible system states. Two tools have been developed for software checking: CBMC (Bounded Model Checker, generally used for embedded systems like nVidia's ASICs) and MAGIC (a counterexample-guided abstraction refinement program). Both have been tested in the field and they have so far proven their worth. Binary Decision Diagrams are used in both tools to represent state transition systems more efficiently.

Computer Science Perspective
Proving that software will work is a desirable outcome for computer science. Software is at the heart of most technology today, including life-and-death systems governing life support and air control. Understanding of how to translate from source code to a formal model that can be tested may also advance the state of computer languages.

Disciplines Actively Involved

Computer Science: This research will improve our understanding of model checking, and while the ramifications of this may include other fields developing the actual method requires only knowledge of Computer Science.

Operational Definition
Actively Involved Disciplines: A Discipline from which a member would be acknowledged in the research paper or any discipline involved for which there is a professional association that fosters more knowledge of it through the scientific method.

Description of Disciplines Involved
Computer Science: The problems of program verification

References Presenter's homepage:
http://www.cs.cmu.edu/~emc/
Additional links:
http://www.cs.cmu.edu/%7Emodelcheck/
http://www.cs.cmu.edu/~modelcheck/tour.htm

By Sduggan (with edits from others)
Updated by plandweh