Discovering The Hard To Find Bugs Using Formal Methods To Find Defects In Hardware & Software Systems

The Utah Verifier Group, School of Computing

Imagine:
Your bank balance is initially $1000. You deposit $500 at an ATM. At the same time, your spouse withdraws $500 at another ATM across town.

Question:
What is your final bank balance?
 a) $500  
b) $1000  
c) $1500

Answer:
Because the bank computer could mix together the instructions from each ATM transaction your final balance could be a, b, or c depending on the order of the instructions.

The Utah Verifier Group:
- Discover difficult to find defects that have severely adverse results.
- Verify system designs against formal correctness specifications.
- Increase overall dependability of computer hardware and software.

Possible Instruction Interleaving

Computer technology, whether it is hardware such as disk drives and sound cards, or software such as Windows and AOL Instant Messenger, is becoming more and more complex. There are also small devices that we don't generally think of when we talk about computers, like the device that controls the anti-lock break system (ABS) in your car, or the circuitry that controls a pace maker. Computers have become part of our world. We depend on them to function correctly every time. If they don't, it will cost us time, money, and perhaps even our lives.

"Formal Verification," is the process of taking a program or a computer part and putting together a formal proof that either the design or the actual product conforms to some specification. Since many of these systems are very complex, a major focus in Formal Verification research has been to make the computer do the work.

My work with the Utah Verifier Group has been on a program that does computerized formal verification. This program is kind of like a rat in a maze. Think of the actions that a computer can perform as being paths in a maze. If every step in every path and every path in the maze conforms to the given specification, then we say that the computer system is "verified" with respect to that specification. This process is called "explicit state model checking." This is what our verification program (rat) does. I have been adding new features to this program to make it possible to explore more complicated and more useful computer systems (mazes).