Record: 1

Send in the Terminator. By: Stix, Gary. Scientific American. Dec2006, Vol. 295 Issue 6, p37-37. 1p. 1 Black and White Photograph. Abstract: The article focuses on Microsoft Research's tool called the "Terminator," which tries to prove that a driver will finish what it is doing. Alan Turing, a mathematician who is among the founders of computer science, created a mathematical proof in 1936 which explained the uncertainty of a computer program to complete a task. INSET: COMPUTER ENTOMOPHOBIA. (*AN: 22953861*)

Persistent link to this record (Permalink):

http://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=22953861&site=ehost-live

Cut and Paste:

<A href="http://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=22953861&site=ehost-live">Send in the Terminator.</A>

Database:

Academic Search Complete

Send in the Terminator

A MICROSOFT TOOL LOOKS FOR PROGRAMS THAT FREEZE UP

Alan Turing, the mathematician who was among the founders of computer science, showed in 1936 that it is impossible to devise an algorithm to prove that any given program will always run to completion. The essence of his argument was that such an algorithm can always trip up if it analyzes itself and finds that it is unable to stop. "It leads to a logical paradox," remarks David Schmidt, professor of computer science at Kansas State University. On a pragmatic level, the inability to "terminate," as it is called in computerese, is familiar to any user of the Windows operating system who has clicked a mouse button and then stared indefinitely at the hourglass icon indicating that the program is looping endlessly through the same lines of code.

The current version of Microsoft's operating system, known as XP, is more stable than previous ones. But manufacturers of printers, MP3 players and other devices still write faulty "driver" software that lets the peripheral interact with the operating system. So XP users have not lost familiarity with frozen hourglasses. The research arm of Microsoft has tried recently to address the long-simmering frustration by focusing on tools to check drivers for the absence of bugs.

Microsoft Research has yet to contradict Turing, but it has started presenting papers at conferences on a tool called Terminator that tries to prove that a driver will finish what it is doing. Computer scientists had never succeeded until now in constructing a practical automated verifier for termination of large programs because of the ghost of Turing, asserts Byron Cook, a theoretical computer scientist at Microsoft Research's laboratory in Cambridge, England, who led the project. "Turing proved that the problem was undecidable, and in some sense, that scared people off," he says.

Blending several previous techniques for automated program analysis, Terminator creates a finite representation of the infinite number of states that a driver could occupy while executing a program: It then attempts to derive a logical argument that shows that the software will finish its task. It does this by combining multiple "ranking functions," which measure how far a device driver has progressed through the loops in a program, sequences of instructions that rerun until a specified condition is met. Terminator begins with an initial, rather weak argument that it refines repeatedly based on information learned from previous failed attempts at creating a proof (a sufficiently strong argument). The procedure may consume hours on a powerful computer until, if everything goes according to plan, a proof emerges that shows that no execution pathway in the driver will cause the dreaded hourglassing.

Terminator, which has been operating for only nine months and has yet to be distributed to outside developers of Windows device drivers, has turned up a few termination bugs in drivers for the soon-to-be-released Vista version of Windows while trying to come up with a proof. Cook predicts that Terminator may eventually find proofs for 99.9 percent of commercial programs that finish executing. (Of course, some programs are designed to run forever.) Turing, however, can still rest in peace. "There will always be an input to Terminator that you can't prove will terminate," Cook says. "But if you can make Terminator work for any program in the real world, then it doesn't really matter."

Patrick Cousot of the École Normale Supérieure in Paris, a pioneer in mathematical program analysis, notes that Terminator should work for a limited set of well-defined applications. "I doubt, for example, that Terminator is able to handle mathematically hard termination problems"--those for floating-point numbers or programs that run at the same time. Cook does not disagree, saying that he plans to develop termination proof methods for such programs. Finding a way to ensure that more complex programs do not freeze is such a difficult challenge, however, that Cook thinks it could consume the rest of his career.

PHOTO (BLACK & WHITE): ALAN TURING created a mathematical proof that explains the uncertainty of any computer program ever completing a task.

~~~~~~~~

By Gary Stix

COMPUTER ENTOMOPHOBIA

Worldwide, software bugs cost billions of dollars In losses every year, which explains a trend among companies for automated program verification. In 2005 Microsoft released an automated bug-catching program, Static Driver Verifier, that checks the source code for device drivers against a mathematical model to determine whether it deviates from its expected behavior.

Static verifiers look for programming errors that cause a program to stop its execution. A device driver, for Instance, should never interact with program B before it has done so with program A, or it will simply cease operation. Terminator, Microsoft's latest tool, looks for mistakes that may lead a program to continue running forever in an endless loop, thereby preventing it from finishing the job at hand.
~~~~~~~~