

Satisfiability: Neue Generation einer statischen Analyse

## Mit Befriedigung auf Fehlerjagd

Embedded Software darf eigentlich keine grüne Banane sein, die erst beim Kunden reift. Die Quellcode-Analyse-Technik der Booleschen Erfüllbarkeit (Boolean Satisfiability – SAT) dient zur Reduktion der Fehlerrate und verbessert so deutlich die Qualität der Software.

Von Ben Chelf \*

Die Entwicklung qualitativ hochwertiger Software ist eine Herausforderung für viele Unternehmen: Denn nur mit innovativen Lösungen können sie kritische Fehler und Schwachstellen in ihren Programmen beheben, bevor die Qualitätssicherung oder, was schlimmer ist, der Anwender im Markt die Defekte entdecken. Lassen sich Software-Patches für Desktop-Rechner und Server noch recht einfach verteilen und aufspielen, so sind Fehler von Embedded Software nur wesentlich aufwändiger auszubügeln. Deshalb bietet eine leistungsfähige Quellcode-Analyse eine echte Erleichterung für die Entwickler von Embedded Software.

Speziell die Kombination von leistungsfähigen Techniken aus dem Bereich der Booleschen Erfüllbarkeit mit Software, die die neuesten Entwicklungen aus der Datenflussanalyse integriert, erzielt durchschlagende Ergebnisse: Sie kann ohne Optimierung – out-of-the-box – eingesetzt, die Fehleraten für »false positive«-Defekte auf unter 10 Prozent drücken und gleichzeitig die Skalierbarkeit auf mehrere zehn Millionen Zeilen C/C++- oder Java-Code erweitern. Eine Analyse kann nur so genau sein, wie die zugrunde liegenden Daten. Deshalb benötigen die Entwickler für die bestmögliche Analyse ihrer Programme ein sehr

genaues Bild von der Software, eine »Software DNA Map«.

Ein Beispiel: Ein Software-Entwicklungsteam erhält ein Software-System mit einigen Millionen Zeilen an Programmcode, ein System also, an dem lange gearbeitet wurde. Das Team soll davon eine Übersicht erstellen, quasi eine Landkarte (Map) der Software, die unter anderem folgende Fragen beantwortet:

- Wie wurde jede einzelne Datei für alle Zielplattformen kompiliert?
- Wie wurden alle Dateigruppen miteinander verbunden?
- Welche verschiedenen Binary-Dateien wurden generiert?
- Welche Funktionen sind in den einzelnen Dateien enthalten, und wie lautet der korrespondierende Aufruf?
- Wie sehen die unterschiedlichen Kontrollflussdiagramme jeder einzelnen Funktion aus?

Es ist praktisch unmöglich, diese Anforderungen manuell zu erfüllen, dafür steht aber inzwischen die automatisierte Lösung »Software DNA Map« zur Verfügung. Sie ist ein Türöffner für statische Analysen und damit für eine erhebliche Verbesserung der Qualität und Sicherheit des Programmcodes. Fehler können schon in frühen Entwicklungsstadien entdeckt werden und so substantielle Vorteile für Software-Entwicklungsfirmen eröffnen. Doch die Performance und Genauigkeit dieser Analysen lassen sich mit der Booleschen Erfüllbarkeit weiter optimieren.

In der Komplexitätstheorie ist das Problem der Booleschen Erfüllbarkeit ein Entscheidungsproblem. Es wird durch einen booleschen Ausdruck definiert, also nur durch UND-, ODER-, NICHT-Operatoren, Variablen und Parenthesen formuliert. Die Frage lautet: Besteht bei einem gegebenem Ausdruck ein Zusammenhang zwischen WAHR- und FALSCH-Werten und Variablen, durch den der Gesamtausdruck WAHR wird? Eine aussagenlogische Formel kann ausreichen, wenn sich ihren Variablen logische Werte so zuordnen lassen, dass das Ergebnis der Formel den Wert WAHR annimmt.

Das Konzept der Booleschen Erfüllbarkeit ist nicht neu, so setzt die EDA-Branche sie bereits in großem Umfang als Unterstützung von Verifikationstechnologien für das Chip-Design ein. Allerdings blieben die benötigten »SAT-Solver« für die Software-Analyse bislang ungenutzt.

Ein SAT-Solver ist ein Computerprogramm mit einer Formel aus Variablen, die durch UND, ODER, NICHT verknüpft sind. Der SAT-Solver determiniert, ob sich die einzelnen Variablen so den Werten WAHR oder FALSCH zuordnen lassen, dass das Ergebnis der gesamten Formel den Wert WAHR annimmt und SAT damit erfüllt ist. Verwirklicht wenigstens eine Zuordnung diese Voraussetzung, gibt der SAT-Solver eine spezielle Erfüllbarkeitsanweisung aus. Wenn nicht, kennzeichnet er die Formel als »nicht erfüllbar«. Zusätzlich kann er die Richtigkeit seiner Entscheidung demonstrieren.

Voraussetzung für die Anwendung von SAT für Software ist, dass der Quellcode in einer Form vorliegt, die automatisch an den SAT-Solver übergeben werden kann. Genau hier kommt die

»Software DNA Map« zum Tragen: Sie stellt die erforderliche Information vom Code in jeder gewünschten Form dar. Weil SAT-Solver mit den Operatoren WAHR, FALSCH, UND, ODER und NICHT arbeiten, können die relevanten Teile des Programms in diese Konstrukte umgewandelt werden.

Zusätzlich ist die Übersetzung der Operationen notwendig, die im Programmcode Teil von Ausdrücken sind. Alle Ausdrücke im Code lassen sich in eine äquivalente Formel transformieren, die mit den Operatoren UND, ODER, NICHT arbeitet. Das ist erforderlich, da ein Compiler die Operationen in Maschinencode-Instruktionen umwandeln muss, die ein Prozessor abarbeiten kann.

Nach der Überführung der gesamten »Software DNA Map« in eine Kombination aus den Booleschen Operatoren lassen sich zahlreiche Formeln daraus ableiten. SAT-Solver können den Code dann analysieren und zusätzliche, komplexere Qualitäts- und Security-Schwachstellen aufspüren. So ist eine erste Anwendung von SAT-Solvern bei der statischen Analyse das »false path pruning«, also das Aufdecken fehlerhafter Pfade. Die Datenflussanalyse meldet manchmal einen Pfad, der während der Laufzeit nicht vorhanden oder unausführbar ist. Da keine passende Variablenkombination für diese Situation im gegebenen Software-System vorliegt, sollte man diesen Fehler bereits anhand der Ergebnisse der statischen Analyse beheben können.

Die Methoden, mit einem solchen Problem innerhalb eines Datenfluss-Frameworks umzugehen, verblissen jedoch gegenüber den Fähigkeiten einer Bit-akkuraten Darstellung im Zusammenspiel mit einem guten SAT-Solver: Denn

\* Ben Chelf ist CTO von Coverity.

