Redundanzarme Generierung von Pfadbedingungen in der Software-Sicherheitstechnik

Torsten Robschink Jens Krinke

18. März 1999

Die Physikalisch-Technische Bundesanstalt (PTB) ist die nationale Oberbehörde im Bereich des gesetzliches Meßwesens und prüft vor der Zulassung alle eichpflichtigen Meßgeräte, wie Käsewaagen, Geschwindigkeitsmeßgeräte, Tankzapfsäulen, Fahrtenschreiber und dergleichen. Derzeit besteht die Zulassung eines Prüflings meist nur aus der Analyse der mechanischen und elektronischen Gegebenheiten und dem Vergleich mit vorhandenen Eichnormalen. Dies ist aber nicht mehr ausreichend, da der Anteil der Software in den Geräten immer mehr steigt.

Die Hersteller haben erkannt, daß eigens für einen einzigen Zweck konstruierte Meßsysteme bei der heutigen Leistungsfähigkeit von Standard-PCs nur unnötig teuer sind, so daß zunehmend software-basierte Systeme auf PC-Basis eingesetzt werden. Diese Universalsysteme können durch Softwaresteuerung jederzeit für ein anderes Einsatzgebiet konfiguriert werden.

Dieses Verfahren ist problematisch, da jedes Meßgebiet unterschiedlichen Vorschriften und Gesetzen unterliegt. Die Zulassung ist mit einem erheblichen Mehraufwand bei der Sicherheits- und Zuverlässigkeitsüberprüfung verbunden, da das meist sachgebietsbezogene Wissen der Prüfer im Bereich Hardware um die Komponente Software erweitert wird. Im Hinblick auf die ständig steigende Komplexität von Software reicht eine einfache Plausibilitätsanalyse der Steuerungssoftware nicht mehr aus, sondern es bedarf leistungsfähiger – und hochkomplexer – Analysesysteme, die den Prüfer unterstützen, ohne ihn zu überfordern.

Im Mittelpunkt der Analyse steht der *Eichpfad*, der den eichpflichtigen Datenfluß vom Meßwertaufnehmer bis zur Meßwertausgabe darstellt. Eine softwaregesteuerte „Beeinflussung“ dieses Pfades kann i.d.R. bei größeren Programmen vom Prüfer nicht mehr durch Code-Review erkannt werden. Hierzu bedarf es maschineller Hilfe, die von unserem Softwareanalyse-Werkzeug VALSOFT geboten wird. Mit diesem Werkzeug kann sich die Zulassungsbehörde rechtlich absichern, indem sie durch elektronische Beweisführung unterstützt wird. VALSOFT versucht, den Quelltext steuernder Software zu analysieren und im Hinblick auf gestellte Fragen (z.B. Einhaltung von Herstellerangaben) zu überprüfen. Wir halten uns dabei strikt an das Prinzip der *konservativen Approximation*: wenn VALSOFT sagt, daß keine Beeinflussung des Eichpfades vorliegt, dann kann sich der Prüfingenieur darauf verlassen. Anderenfalls kann die VALSOFT-Ausgabe als juristisch relevantes Indiz für den beanstandeten Datenfluß dienen.

Wir übertragen den ANSI-C-Quelltext der Software in eine für Analysezwecke geeignete Struktur, dem Programmabhängigkeitsgraphen. Er stellt die Daten- und Kontrollabhängigkeiten des Programmes dar und spiegelt somit das statische Verhalten des Systems wider. Beeinflussungen von Eichpfaden können durch sog. *Slices* ermittelt werden: Slices zu einem bestimmten Punkt im Programm enthalten diejenigen Anweisungen, die den Punkt in irgendeiner Weise beeinflussen können.

Mit Slices kann nur festgestellt werden, ob eine Beeinflussung möglich ist, aber nicht, unter welchen Bedingungen. Daher ermitteln wir genaue Bedingungen, unter denen „illegale“ Datenflüsse stattfinden. Diese sog. *Pfadbedingungen* stellen notwendige Bedingungen für die Ausführung von Pfaden zwischen Programmpunkten dar und können nach der Auflösung durch externe Constraint-Solver automatisch Aussagen über tatsächliche Datenflüsse liefern.

Für kleine Programme konnten wir mit unserem bisherigen Verfahren erfolgreich Pfadbedingungen aufstellen. Bei größeren Programmen wurden die Bedingungen jedoch immens groß, ließen sich nur ineffizient berechnen und enthielten zu viel Redundanz. Das vorgestellte Verfahren schafft Abhilfe, indem es den Aufbau dieser Bedingungen beschleunigt und deutlich die auftretende Redundanz vermindert. Da die Bedingungen über Graphen aufgestellt werden, können wir die Eigenschaften von *Artikulationspunkten* und *starken Zusammenhangskomponenten* ausnutzen. Im Zusammenhang mit einer heuristischen Untergraphberechnung wird der zugrundeliegende Graph soweit wie möglich in unabhängige Einheiten zerlegt, die anschließend einzeln ausgewertet werden können.

Wir nutzen dabei den Ansatz, Knoten zusammenlaufender Pfade innerhalb des Graphen schon zu Beginn des Pfadbedingungsaufbaus zu erkennen und dadurch Redundanz vom Anfang an zu vermeiden. Der Graph wird an Artikulationspunkten in disjunkte Teilgraphen zerlegt, über denen unabhängig voneinander Pfadbedingungen gebildet werden können. Auftretende Zyklen werden dynamisch zu einzelnen Knoten gefaltet. Durch dieses Verfahren erhalten wir eine geordnete Kette von disjunkten Teilgraphen.

Eine heuristische Untergraphbestimmung versucht anschließend, diese Teilgraphen an wohldefinierten Stellen aufzubrechen und neue Untergraphen zu bilden, die wiederum auf zusammenlaufende Pfade untersucht werden. Das Verfahren beruht dabei auf der Erfahrung, daß sich unter Hinnahme einer geringen Redundanz im ersten Schritt um so mehr Redundanz in den folgenden Schritten vermeiden läßt.

Dieses Verfahren vermeidet die Explosion der Bedingungen und ermöglicht es, sowohl die Berechnung zu beschleunigen als auch größere Programme zu analysieren.