Lightweight Verification 2008

Martin Leucker, Helmut Seidl

Institut für Informatik
Technische Universität München
Boltzmannstraße 3
85748 Garching
seidl@in.tum.de
leucker@in.tum.de

1 Übersicht

Ziel des Workshops ist es, praxistaugliche Verifikationstechniken, die insbesondere zum Auffinden von Fehlern geeignet sind, den Teilnehmern näher zu bringen. Dazu stellen drei eingeladene Sprecher neueste Ergebnisse in den jeweils von ihnen vertretenen Gebieten vor.

Der Workshop richtet sich nicht nur an Wissenschaftler, die in diesem Bereich arbeiten, sondern insbesondere auch an Praktiker, die sich hier über neue Verifikationstechniken informieren wollen.

2 Beiträge

- **Ernst-Rüdiger Olderog - Kollisionsfreiheit von Verkehrsagenten**

  Die Kollisionsfreiheit von Verkehrsagenten wie Autos, Bahnen und Flugzeuge ist eine zentrale Sicherheitsanforderung im Transportbereich. Eine Verifikation dieser Anforderung ist schwierig, da es sich bei den Verkehrsagenten um hybride, d.h. diskret-kontinuierliche Systeme handelt.

  In diesem Vortrag wird aus einem Design-Pattern für Verkehrsagenten eine Beweisregel gewonnen, die den Nachweis der globalen Eigenschaft der Kollisionsfreiheit auf Prämissen reduziert, die einfacher und teilweise automatisch beweisbar sind. Die Regel basiert auf den Konzepten einer Sicherheitsumgebung und einer Kritikalitätsfunktion.

- **Andreas Podelski - Terminierungsanalyse für Reaktive Systeme**

  Reaktive Systeme (wie z.B. Betriebssysteme, Web Server, Mail Server, Datenbank-Maschinen, etc.) sind gewöhnlich aufgebaut von einer Menge von Komponenten, von denen wir erwarten, dass ihre Aufrufe immer terminieren. Tatsächlich ist die Terminierung eine zentrale Bedingung dafür, dass das System reaktionsbereit bleiben kann.

  Obwohl Terminierung und Terminierungsanalyse seit langem Gegenstand theoretisch orientierter Untersuchungen sind, hat die Forschung an praktischen Methoden zur Terminierungsanalyse von realistischen Programmen erst vor einigen Jahren begonnen. Wir berichten von laufenden Entwicklungen solcher Methoden und ihrer Anwendung für reaktive Systeme.

- **Andreas Zeller - Mining temporal program properties**

  When using an API, programmers not only need to take care of issuing the correct calls with the correct arguments. Often, the individual services of an API also must be invoked in a specific order; likewise, the arguments are typically constructed using specific API calls.

  Our tools mine program code and program executions to derive typestate-like models of such call sequences for objects and parameters. Using such models, we uncover real-life bugs in real-life programs like AspectJ bugs where the sequence of construction methods for arguments deviates from common (and correct) usage.