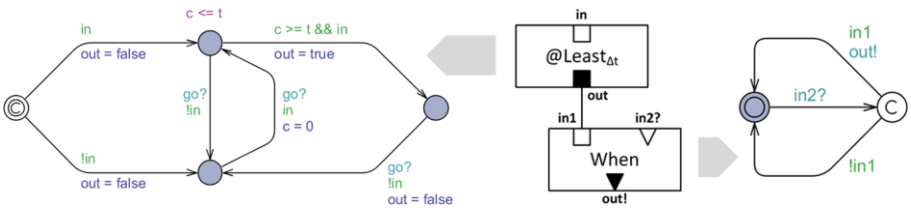
Abschlussarbeit (BA/MA)

Spezifikationsanalyse für Echtzeitsysteme

Thema

Die meisten Modellprüfungsansätze erfordern die Existenz eines operationellen Systemmodells, um es gegen formale Anforderungen zu verifizieren. Wenn jedoch das Hauptziel darin besteht, sicherzustellen, wie es in frühen Phasen der **Software-Entwicklung für eingebettete Systeme** oft der Fall ist, dass eine formale Systemanforderung und ihre Aufteilung in Komponentenanforderungen (als sogen. Observer-Automaten) konsistent sind, kann dies auf Grundlage der Anforderungen allein geschehen. Diese Art der Spezifikationsintegration bezeichnen wir als **„implementierungsfreie Verifikation“**. Sie ist verwandt mit dem Problem des verteilten Testens, also mit der Frage, wie sich globale Eigenschaften lokal testen lassen. Eine solche Methode eignet sich insbesondere für die **frühzeitige Validierung eines Software-Designs**, bevor eine Implementierung vorliegt.

Zu diesem Zweck entwickelt unser Lehrstuhl ein Model-Checking-Framework auf der Grundlage von Netzen von Timed Automata (TA) sowie einer dedizierten Beschreibungssprache für diese (s. Abbildung). Eine Besonderheit unseres Ansatzes ist die gleichwertige Berücksichtigung von **Ereignissen/Transitionen *und* Signalen/Zuständen** in einer Spezifikation. TA sind ein verbreiteter Formalismus zur Verifikation von Echtzeitsystemen, jedoch lassen sich viele Abstraktionsmethoden auf Anforderungsebene nicht ohne weiteres einsetzen. Andererseits bieten die Eigenschaften der Spezifikationssprache (Determinismus, Unterscheidung von Eingangs- Ausgangs- und internen Variablen, bedingte Spezifikation mit Assumptions und Commitments, u.a.m.) weitere Möglichkeiten der Effizienzsteigerung.



SOFTWARE-SPEZIFIKATION MIT NETZEN STANDARDISIERTER TA-FRAGMENTE

Wir bieten mehrere Abschlussarbeiten zu diesem Thema an, die das gemeinsame Ziel haben, **skalierbare Techniken** zur Exploration des Zustandsraums von TA-basierten Spezifikationen zu entwickeln sowie die **Fehlersuche** darin zu beschleunigen, und damit den Einsatz formaler Software-Spezifikation und -Verifikation in der industriellen Praxis zu befördern.

Aufgaben & Ziele

Eine Bachelor-Arbeit zu diesem Thema umfasst beispielsweise

- den Vergleich existierender Methoden zur Zustandsraumexploration von TA-Netzen; Definition einer Fallstudie,
- die Entwicklung eines Explorationsansatzes für TA-Netze aus standardisierten Fragmenten,
- einen Leistungsvergleich des selbsterarbeiteten Ansatzes mit Model-Checkern wie z.B. UPPAAL, OCRA oder Rabbit.

Eine Masterarbeit zu diesem Thema beschäftigt sich darüberhinaus mit ausgewählten Erweiterungen, z.B.

- Integration eines Komponentenkonzepts; Halbordnungsreduktion mit Entfernung verschwindender, zeitloser Zustände,
- Entwicklung von Heuristiken für gerichtetes („guided“) Model-Checking auf TA-Observern,
- theoretische Untersuchung von Aspekten/Grenzen der Spezifikationssprache (Mächtigkeit, Entscheidbarkeit u.ä.),
- Plausibilitätsprüfungen: Realisierbarkeit, Trivialität, Redundanz von (Teil-)Spezifikationen.

Diese Aufzählung gibt Anhaltspunkte für mögliche Inhalte; die konkrete Aufgabengestaltung ist Verhandlungssache.

Vorkenntnisse

Für Interessent(inn)en sind Vorkenntnisse auf folgenden Gebieten hilfreich: (1) Programmierung und Unit-Testing mit Java und Eclipse, (2) Automatentheorie, Temporallogik, Model-Checking, (3) XML, Parser-Generierung. Fehlende Vorkenntnisse in manchen Punkten lassen sich durch Begeisterung, Lernbereitschaft und Zielorientierung kompensieren.

Literatur

T. Launiainen, K. Heljanko, T. Junttila (2011). Efficient model checking of PSL safety properties. *IET Comput. Digit. Tech. 5(6)*.
N. Lynch, L. Michel, A. A. Shvartsman (2008). Tempo: a toolkit for the TIOA formalism. Technical report, MIT/UConn.
J. Bengtsson, W. Yi (2004). Timed automata: semantics, algorithms and tools. *LNCS 3098*.

Ansprechpartner

Marc Förster, MSc
foerster@embedded.rwth-aachen.de