Die Klassifikationsbaummethode für eingebettete Systeme mit Testmustern für nichtkontinuierliche Reglerelemente

Alexander Krupp, Wolfgang Mueller
Universität Paderborn/C-LAB, Paderborn, Germany

Abstract: Ein wichtiges Qualitätsmerkmal des modellbasierten Tests und damit beim Test von Simulink-Modellen sind Überdeckungskriterien. Wir stellen einen Ansatz zur Verbesserung der Testüberdeckung von Simulink-Modellen unter Verwendung der Klassifikationsbaummethode (CTM) vor. Dazu erweitern wir die CTM um eine Signalbeschreibung durch die Ableitung eines Signals.

1 Einleitung

Insbesondere die frühen Phasen des Entwurfs elektronischer Systeme beinhalten mehrere Herausforderungen. Anforderungen werden hier meist in natürlichsprachliche Sätze in Tabellenstrukturen gefasst, die die Grundlage für den Entwurf und Analyse des zu entwerfenden Systems bilden. In weiteren Analysen werden formale Verifikationen [Kro99] und Simulationen [BCNN04] eingesetzt. Simulationen werden immer auf Basis eines Modells durchgeführt. Je nach Anwendungsbereich und Modellierungstiefe bzw. Abstraktionsgrad finden hier verschiedene Simulatoren z.B. Matlab/Simulink, SystemC, SystemVerilog, oder VHDL ihre Anwendung. Während die Verfahren zur automatischen Testmuster-Generierung für spätere Entwurfsphasen ziemlich ausgereift sind [JG03, Cro99], ist die Testmuster-Erstellung in frühen Phasen ein noch relativ unbearbeitetes Gebiet und wird meist manuell durchgeführt. In jüngster Zeit findet hier die Idee des modellbasierten Tests größere Beachtung, wobei Informationen zur Erstellung der Testmuster aus der Struktur des jeweiligen Modells gewonnen werden. Zur strukturierten Ableitung modellbasierter Tests schlägt Conrad [Con04] Klassifikationsbäume als Zwischenschritt vor. Ein Klassifikationsbaum wird manuell aus der Anforderungsbeschreibung und dem Modell erstellt und dient zur automatischen Ableitung von Testumgebungen mit denen das simulationsfähige Modell validiert wird. Im Automobilbereich liegt das Modell häufig als Matlab/Simulink-Modell vor. Verschiedene nichtkontinuierliche Regelelemente erweitern einen Regler um zusätzliche Arbeitsbereiche, die bei Regleranalyse und –test berücksichtigt werden müssen. In Simulink wurden deshalb Überdeckungskriterien (wie z.B. Decision- oder Condition-Coverage) für nichtkontinuierliche Modellelemente eingeführt[Ald02].

Wir fügen der Klassifikationsbaummethode für eingebettete Systeme eine neue Art der Signalrepräsentation hinzu. Zusätzlich zur abstrakten Signalverlaufsdefinition läßt sich damit die Ableitung eines Signals über Äquivalenzklassen angeben. Durch die Verwendung des Dirac-Impulses bei der Beschreibung der Ableitung lassen sich so auch Sprunghöhen eines Signals unabhängig von dessen aktuellem Wert klassifizieren. Diese Eigenschaft läßt

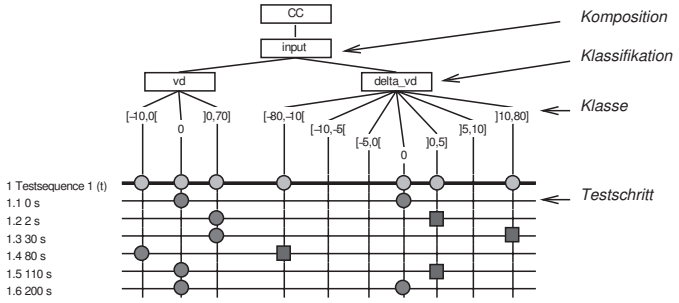

Abbildung 1: CTE Baum für vd, Δvd

sich z.B. in einem White-Box-Test ausnutzen, um eine höhere Testüberdeckung für bestimmte nichtkontinuierliche Reglerelemente in Simulink zu erreichen.

2 Klassifikationsbäume und der Dirac-Impuls

Zur strukturierten Darstellung von Testfällen wurden Anfang der 90er Jahre bei der Daimler–Benz AG die Klassifikationsbäume entwickelt [GG93]. Die Erstellung von Klassifikationsbäumen und der Kombinationstabelle wird durch die auf der CP-Methode (Category-Partition) basierende Klassifikationsbaummethode unterstützt [OB88]. In der ursprünglichen Ausprägung beschreiben Klassifikationsbäume verschiedene Eingabekonstellationen für ein Testobjekt. Um auch zeitabhängige Testverläufe formulieren zu können, wurde die Notation seit 1998 schrittweise zur Klassifikationsbaummethode für eingebettete Systeme (CTM/ES) erweitert [Con04]. Der Klassifikationsbaum (siehe Abb. 1) wird speziell aus der technischen Schnittstelle des Testobjektes abgeleitet, d.h. jede Eingangsgröße des Testobjektes (hier vd) wird durch eine *Klassifikation* dargestellt. Der zulässige Wertebereich einer Eingangsgröße wird dann über Intervalle oder Einzelwerte in *Klassen* partitioniert.

Eine abstrakte Testsequenz wird aus einzelnen Testschritten aufgebaut, die als zeitliche Abfolge in die Zeilen der Kombinationstabelle abgebildet werden. Der Aktivierungszeitpunkt eines Testschritts wird als Synchronisationspunkt bzw. als Stützstelle bezeichnet. Der Zeitpunkt wird in einer zusätzlichen Spalte zu jeder Stützstelle angegeben. Der Signalverlauf zwischen den Stützstellen wird durch verschiedene Interpolationen beschrieben, wobei der Übergang z.B. als Sprung–, Rampen–, oder Sinusfunktion erfolgen kann. In der Kombinationstabelle werden Übergangsfunktionen durch verschiedene Linientypen zwischen den Stützstellen repräsentiert.

Dabei stellen die verschiedenen Klassifikationen mit ihrem Wertebereich X_i und die zugeordneten Klassen $P_i(X_i)$ der CTM einen abstrakten Zustandsraum der Eingabevariablen dar. Jeder Eingabevariablen des Testobjekts entspricht eine Klassifikation. Der zeitliche Ablauf eines Testszenarios wird mit Hilfe von Stützstellen beschrieben. Wenn $T = \{t_0, t_1, ..., t_e\}$ mit $t_0 < t_1 < ... < t_e$ die Menge der Stützstellen eines Testszenarios ist, so heißen die Zeitintervalle $[t_0, t_1], [t_1, t_2], ..., [t_{e-1}, t_e]$ Testschritte. Eine Wer-

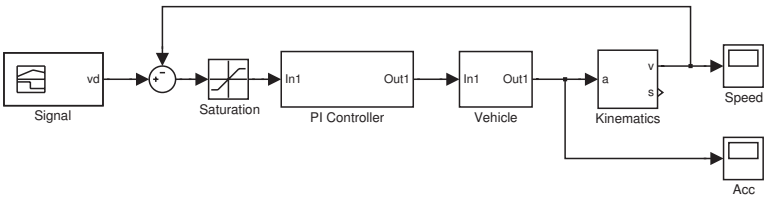

Abbildung 2: Simulink Fahrtregler

tefunktion $v_i : T \rightarrow X_i$ ordnet jeder Stützstelle einen Wert zu. Eine Wertefunktion v_i heißt *kompatibel*, wenn zu jedem Zeitpunkt t gilt, daß $v_i(t)$ zu der entsprechenden Klasse gehört. Eine Interpolationsfunktion $i_i = T \rightarrow I$ mit $I = \{step, ramp, sine, ...\}$ ordnet der Eingangsvariable i zu jedem Zeitpunkt aus T eine Interpolationsvorschrift zu. Die Interpolationsvorschrift $i_{i,t_k}(t)$ gibt den Wert einer Eingangsvariablen ab der Stützstelle t_k an. Bei Verwendung von Rampenfunktion und Sinus-Halbwelle als Interpolationsvorschrift ergibt sich ein stetiger Testdatenverlauf $\bar{v}_i : \bar{T} \rightarrow X_i$, wobei $\bar{T} \supset T$ eine strikt geordnete Menge von Zeitpunkten im Sinne des klassischen Zeitbegriffs ist.

Angelehnt an [Con04] schlagen wir eine Erweiterung der Klassifikationsbaummethode CTM/ES vor, die die vorhandenen Interpolationsmöglichkeiten erweitert und so die Klassifikation von Sprunghöhen einer Eingabevariable gestattet. D.h. bei mehrfacher Verwendung der Interpolation $step$ in einer Testsequenz läßt sich die Sprunghöhe über eine Klassifikation vorgeben. Dazu beschreiben wir den Testdatenverlauf zusätzlich über seine Ableitung 1. Ordnung, und verwenden die Dirac-Distribution $\delta(t)$ und die Heaviside'sche Sprungfunktion $H(t)$ [Föl94]. Sei $\Delta v_i : T \rightarrow Y$ die zur Wertefunktion v_i gehörige, analog zu v_i über Stützpunkte definierte 1. Ableitung. Δv_i heiße Differenzwertefunktion mit dem Wertebereich Y. Sei $\overline{\Delta v_i} : \bar{T} \rightarrow Y$ deren zeitkontinuierliche Fortsetzung:

$$\overline{\Delta v_i}(t) = \sum_{j=0}^{e-1} \begin{cases} \Delta v_i(t_j)\delta(t - t_j) & i_i(t_j) = \text{dirac} \\ i_{i,t_j}(t)(H(t - t_j) - H(t - t_{j+1})) & i_i(t_j) \neq \text{dirac} \end{cases} \quad (1)$$

wobei i_{i,t_j} die Interpolationsvorschrift ist und i_i die Interpolationsfunktion. Ist die Interpolationsvorschrift keine Dirac-Distribution, so entspricht der kontinuierliche Verlauf von $\overline{\Delta v_i}(t)$ der üblichen Interpolationsvorschrift [Con04]. An jeder Stützstelle t_j, die die Dirac-Distribution verwendet, wird die Sprunghöhe $\Delta v_i(t_j)$ aus dem Klassifikationsbaum und der Kombinationstabelle ermittelt. Der Testdatenverlauf ergibt sich durch Integration von $\overline{\Delta v_i}(t)$ und mit Hilfe des Startwerts $v_i(t_0)$.

3 Fallbeispiel Tempomat

Bereits seit den 80er Jahren wird der Tempomat als Komfortfunktion in Kraftfahrzeugen angeboten. Eine weiterentwickelte Variante mit Sicherheitsfunktion wird seit 1998 angeboten[1]. In Abbildung 2 ist ein Regelkreis in Anlehnung an den Regelkreis für hinder-

[1] z.B. von DaimlerChrysler als "DISTRONIC"

nisfreie Fahrt eines solchen Tempomaten in Simulink-Notation abgebildet. Die Differenz zwischen Soll- und Ist-Geschwindigkeit ($vd - v$) des Fahrzeugs soll ausgeregelt werden. Die Regeldifferenz, die dem PI–Regler zugeführt wird, ist in diesem Modell durch eine Sättigung begrenzt. Hinter dem PI–Regler folgt die modellierte Strecke aus der die aktuelle Geschwindigkeit v wieder rückgeführt wird. Ein nichtkontinuierliches Regelelement wie die hier verwendete Sättigung erweitert den Regler um zusätzliche Arbeitsbereiche, die bei Regleranalyse und –test berücksichtigt werden müssen, um eine entsprechende Testüberdeckung zu erzielen. In dem hier gezeigten einfachen Fall erfolgt die Sättigung jeweils bei den Werten 5 und -5.

Der Klassifikationsbaum zum Testobjekt "Tempomat" ist in Abb. 1 zu sehen. Der einzige Eingang vd ist als Klassifikation im Klassifikationsbaum abgebildet. Eine weitere Klassifikation bezieht sich auf die Ableitung Δvd von vd. Zu jeder Klassifikation ist eine Menge von Klassen definiert, die den Wertebereich der Klassifikationen mit Hilfe von Intervallen überdecken. Der Wertebereich für vd ist $X_{vd} = [-10, 70]\,\frac{\mathrm{m}}{\mathrm{s}}$. Der Wertebereich $Y_{\Delta vd} = [-80, 80]\,\frac{\mathrm{m}}{\mathrm{s}^2}$ ist so aus dem Umfang von X_{vd} ermittelt, daß ein durch Δvd vorgegebener Sprung von vd maximal über das gesamte Intervall X_{vd} hinweg erfolgen kann. Der Wertebereich jeder Klassifikation wird hierbei in Klassen unterteilt. Für die Geschwindigkeit *v* gibt es 3 Klassen: $\{[-10, 0[,\ 0,\ ]0, 70]\}$. Für Δvd sind 7 Klassen definiert. Diese wurden ermittelt unter der Annahme eines stationären Zustands von $vd - v$. Die 3 Klassen $[-5, 0[,\ 0,\ ]0, 5]$ beinhalten diejenigen Werte für Δvd, die nicht zum Eintritt in den Sättigungsbereich genügen, d.h. $vd - v \in [-5, 5]$. Die beiden Klassen $[-10, -5[,\ ]5, 10]$ genügen, um möglicherweise den Sättigungsbereich zu erreichen, wenn $(vd - v) \in [-5, 5]$. Die beiden letzten Klassen $[-80, -10[,\ ]10, 80]$ genügen immer um in die Sättigung zu gelangen. Die Bestimmung des Klassifikationsbaumes aus dem Systeminterface wird in [Con04] beschrieben. Die zusätzliche Bestimmung der Klassen der Differenzwertefunktion erfolgt automatisch aus dem Simulink-Modell, unter Berücksichtigung nichtkontinuierlicher Eingangselemente. Mit Hilfe des danach vorliegenden Klassifikationsbaums wird anschließend eine Testsequenz erzeugt. Eine Testsequenz, die alle Klassen von Δvd mit dem Dirac-Impuls überdeckt, und die nach jeder Anwendung eines Impulses auf Δvd (und damit eines Sprungs auf vd) dem Regler ausreichend Einschwingzeit gewährt, erzielt mit hoher Wahrscheinlichkeit eine vollständige Überdeckung für das betrachtete Sättigungsglied. In dem Klassifikationsbaum in Abb. 1 wurde eine Testsequenz über die Kombinationstabelle definiert, die zwar nicht alle Klassen von Δvd überdeckt, die aber bereits für eine vollständige *Decision Coverage* in Simulink genügt. Das bedeutet, daß alle drei Abschnitte der Sättigungsfunktion mindestens einmal während des Tests erreicht werden. Die Instantiierung zu einer eindeutigen Testsequenz erfolgt automatisch, z.B. durch zufällige Auswahl der Stützstellen, o.ä. [BCS97, BN03]. Eine solche Testsequenz ist in Abb. 3 zu sehen.

4 Zusammenfassung

In dieser Arbeit stellen wir einen Ansatz zur Verbesserung der Testüberdeckung von Simulink-Modellen unter Verwendung der Klassifikationsbaummethode für eingebettete

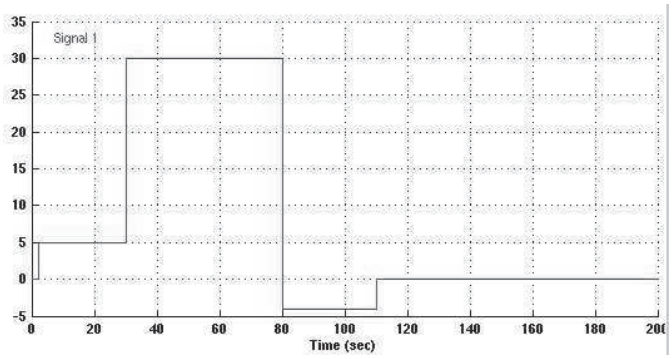

Abbildung 3: Simulink Signal vd

Systeme (CTM/ES) vor. Dazu erweitern wir die CTM um die Signalbeschreibung durch ihre Ableitung, die u.a. durch Dirac–Impulse beschrieben werden kann. Bei gleichzeitiger manueller Definition des Signals und seiner Ableitung können jedoch leicht Inkonsistenzen auftreten, die durch entsprechende Maßnahmen abzuprüfen sind. Bei der Erstellung der konkreten Testsequenz sind ebenfalls Inkonsistenzen möglich. Zur praktischen Anwendung dieser Methode der Testsequenzgenerierung erscheint weitere Werkzeugunterstützung daher unverzichtbar.

Danksagung

Die hier beschriebene Arbeit wurde durch das BMBF-Projekt IMMOS gefördert. Wir bedanken uns außerdem bei Mirko Conrad von DaimlerChrysler, sowie bei Otto Linnemann und Klaus Lamberg von dSPACE für viele hilfreiche Anregungen und Diskussionen.

Literatur

[Ald02] William Aldrich. Using Model Coverage Analysis to Improve the Controls Development Process. AIAA GN&C, 2002.

[BCNN04] J. Banks, J. Carson, B. L. Nelson und D. Nicol. *Discrete-Event System Simulation*. Prentice Hall, 4. Auflage, 2004. ISBN 0-131-44679-7.

[BCS97] Standard for Software Component Testing. Working Draft 3.3. Bericht, British Conputer Society (BCS), Specialist Interest Group in Software Testing, April 1997.

[BN03] E. Broekman und E. Notenboom. *Testing Embedded Software*. Addison-Wesley, London(GB), 2003.

[Con04] Mirko Conrad. *Modell-basierter Test eingebetteter Software im Automobil*. Deutscher Universitäts-Verlag, Wiesbaden, 2004.

[Cro99] Alfred Crouch. *Design-For-Test For Digital IC's and Embedded Core Systems*. Prentice Hall PTR, 1999. ISBN 0-130-84827-1.

[Föl94] Otto Föllinger. *Regelungstechnik*. Hüthig, 8. Auflage, 1994.

[GG93] Matthias Grochtmann und Klaus Grimm. Classification Trees for Partition Testing. *Softw. Test., Verif. Reliab.*, 3(2):63–82, 1993.

[JG03] N. K. Jha und S. Gupta. *Testing of Digital Systems*. Cambridge University Press, 2003. ISBN 0-521-77356-3.

[Kro99] Thomas Kropf. *Introduction to formal hardware verification*. Springer, 1999. ISBN 3-540-65445-3.

[OB88] Thomas J. Ostrand und Marc J. Balcer. The Category-Partition Method for Specifying and Generating Functional Tests. *Commun. ACM*, 31(6):676–686, 1988.