Eingrenzung möglicher Ursachen von Laufzeitfehlern in Simulink-Modellen

Johanna Schneider – Daimler AG – 09. Juli 2014
johanna.schneider@daimler.com

Mercedes-Benz

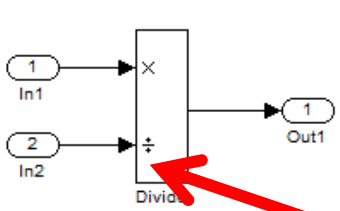
Agenda

- Motivation
- Ursachen von Laufzeitfehlern in Simulink-Modellen
  - Statistisches Lernen
  - Beispiel
  - Herausforderungen
- Ergebnisse und Ausblick

Motivation

- Modellbasierter Entwicklungsprozess
- Code muss fehlerfrei sein (funktional als auch bzgl. Laufzeitfehler)

  → Code-Analyse (hier: Polyspace um Laufzeitfehler zu finden)
- Defekt im Code gefunden:
  - Korrektur im Code → Defekt bei Wiederverwendung des Modells weiterhin enthalten
  - Korrektur im Modell



Code-Generierung

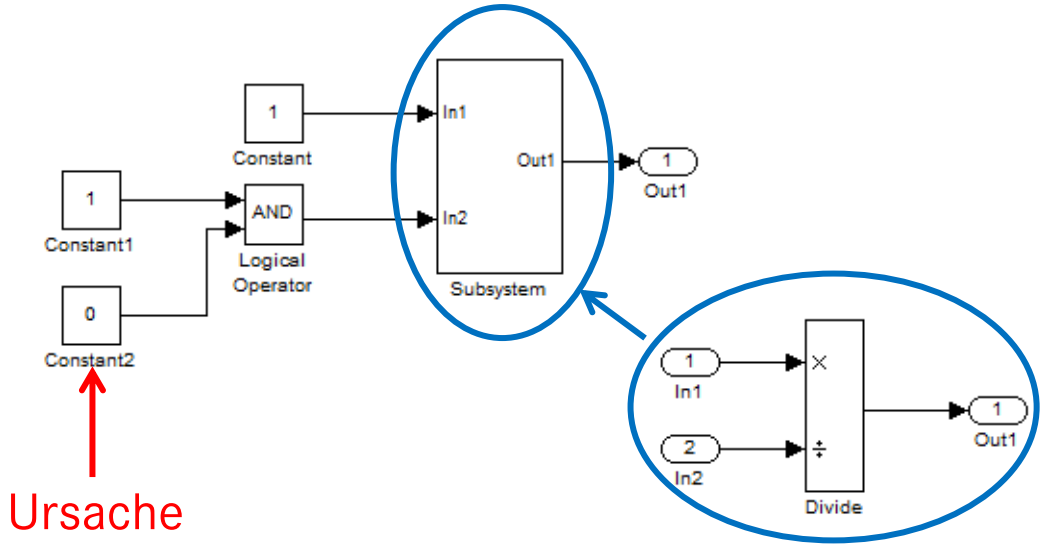
```
Out1 = In1/In2;
```

Auf Steuergerät laden

Division durch Null möglich

Motivation

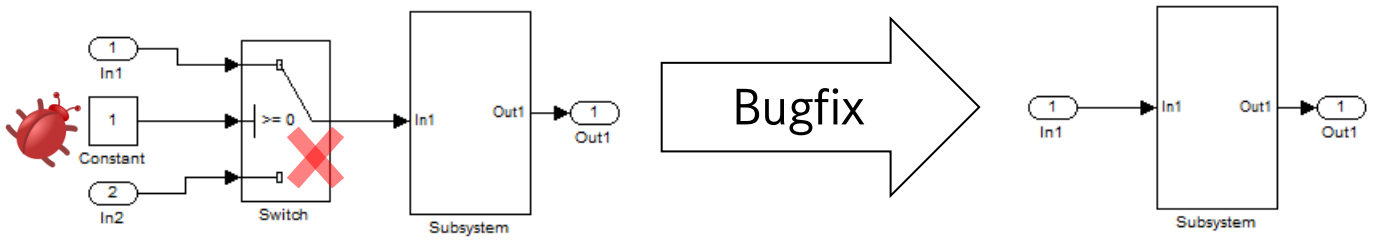
- Bevor ein Defekt korrigiert werden kann muss bekannt sein, wodurch er entsteht → Ursache
- Ziel: Ursachen für gemeldete Defekte semi-automatisiert im Modell finden
  - Statistisches Lernen aus vorangehenden Korrekturen von Laufzeitfehlern



- **Reduzierung des Reviewaufwands um 99,45 %**

Statistisches Lernen

- Einmalig pro Defekt-Typ (Dead Code, Array out of bounds,...)
- Lernen anhand von bereits analysierten Modellen



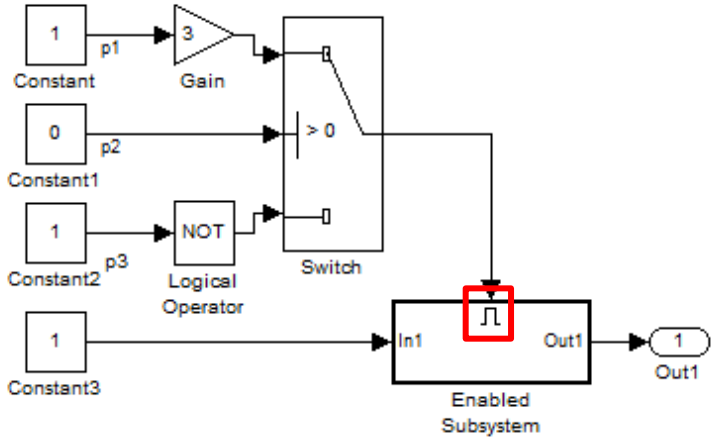
Version A

Version B

- Statistik, welcher Blocktyp (Constant, Inport, Switch,...) wie oft an einem Defekt beteiligt ist

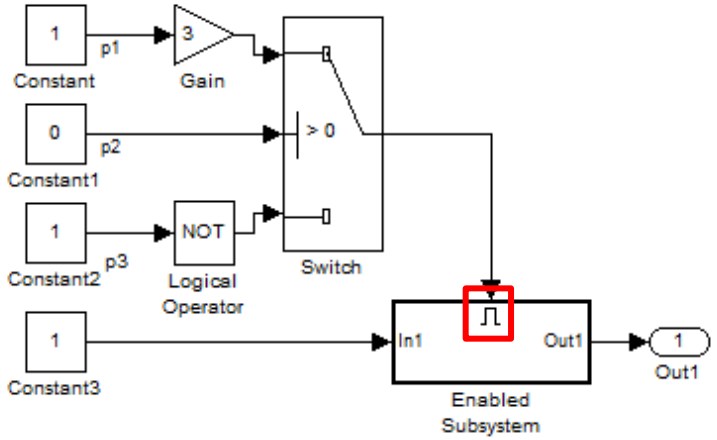
Beispiel

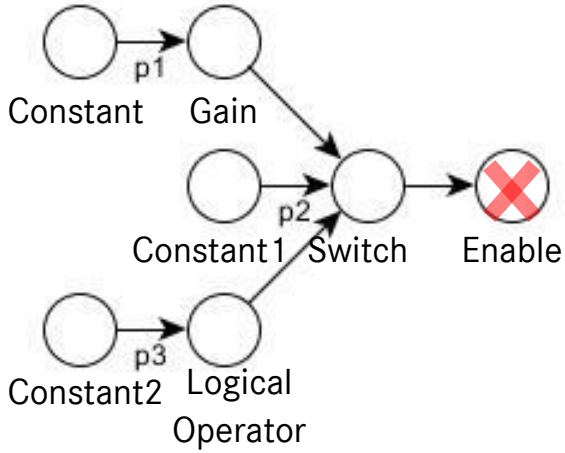
- Mapping eines Code-Defekts auf das Simulink-Modell



Beispiel

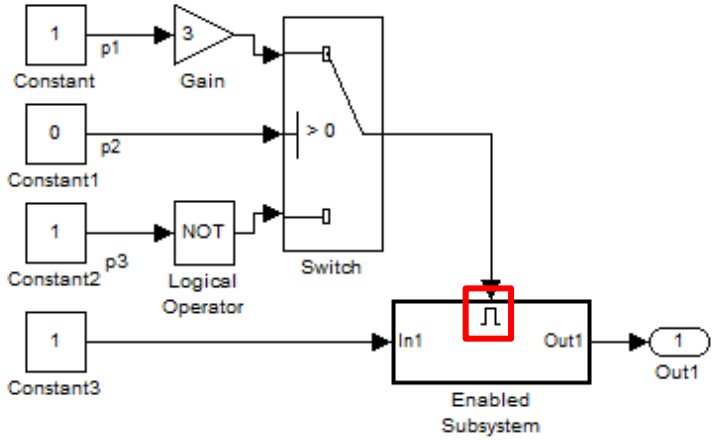
- Mapping eines Code-Defekts auf das Simulink-Modell
- Alle Pfade, die zum Defekt führen

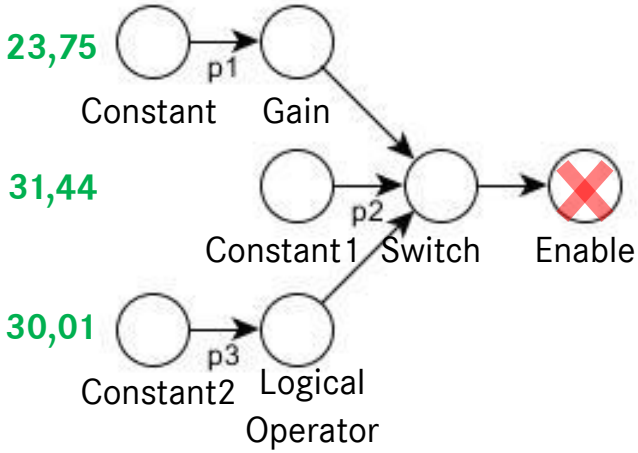




Beispiel

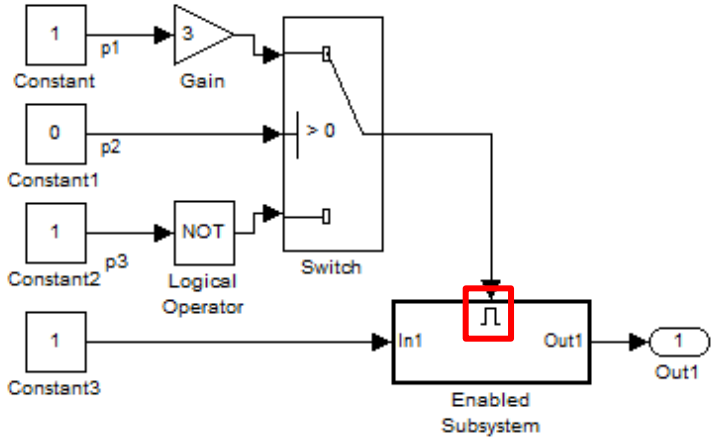
- Mapping eines Code-Defekts auf das Simulink-Modell
- Alle Pfade, die zum Defekt führen
- Gewichtung der Pfade anhand des statistischen Lernens

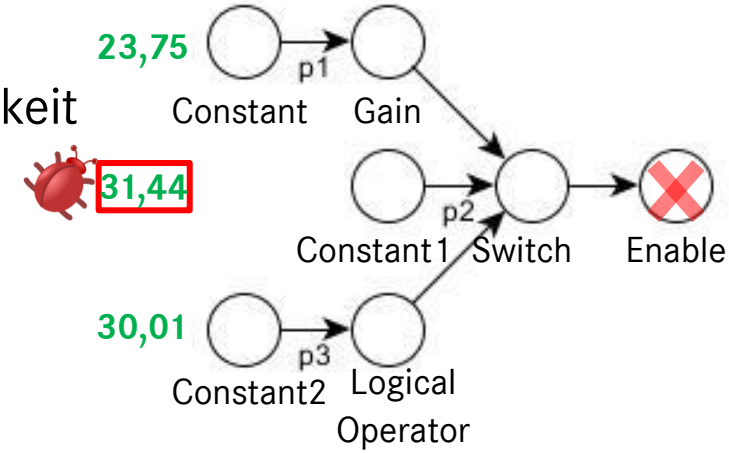




Beispiel

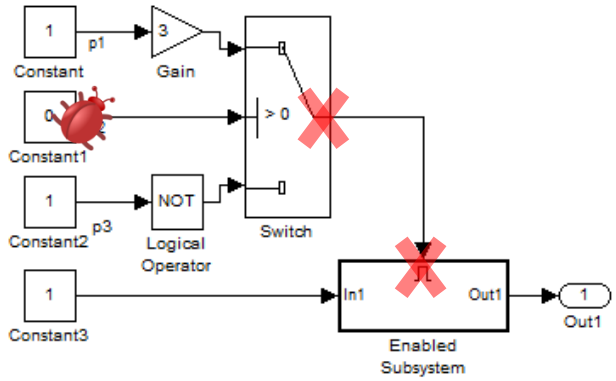
- Mapping eines Code-Defekts auf das Simulink-Modell
- Alle Pfade, die zum Defekt führen
- Gewichtung der Pfade anhand des statistischen Lernens
- Höheres Gewicht ⇔ höhere Wahrscheinlichkeit die Ursache zu enthalten





Herausforderungen

- Mögliche Abhängigkeiten zwischen den Defekten
  - Mehrere Defekte haben die selbe Ursache



- Ursache muss nicht eindeutig sein
- Ursache kann über mehrere Pfade verteilt sein
- Mapping Code ↔ Modell nicht immer eindeutig
- Heuristisches Verfahren → das Verfahren muss nicht immer erfolgreich sein

Ergebnisse

- 12 Simulink-Modelle
  - Zwischen 1.000 und 22.000 Blöcke je Modell
  - 129 Defekte vom Typ Dead Code
  - Ca. 26.000 Pfade insgesamt gefunden mit 1 - 5.000 Pfade pro Defekt

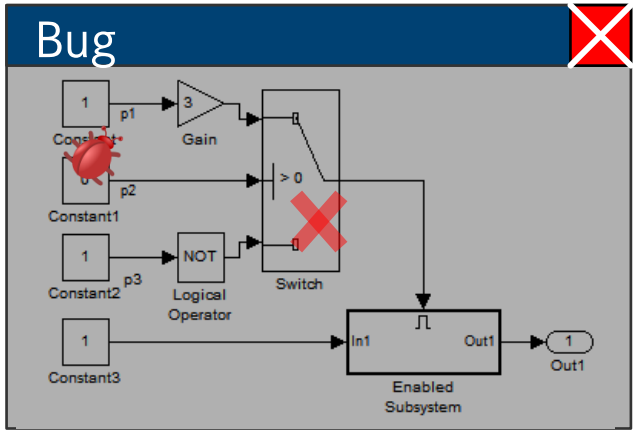
Pfade mit:	# gereviewter Pfade	# gefundener Ursachen
Höchstem Gewicht	129	120/129 (93 %)
Zweithöchstem Gewicht	129 + 9 = 138	124/129 (96%)
Dritthöchstem Gewicht	138 + 5 = 143	129/129 (100%)

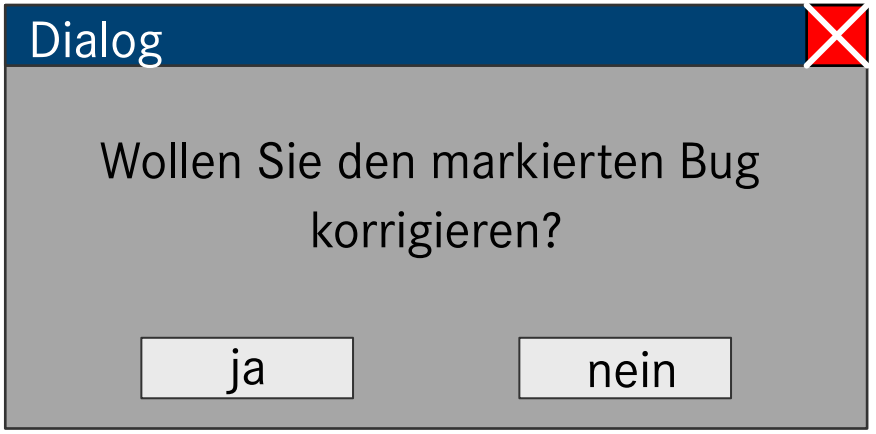
- **Insgesamt: 143 von 26.000 Pfaden mussten gereviewt werden**

→ 0,55% des eigentlichen Reviewaufwands

Ausblick

- Anwendung des statistischen Lernens auf weitere Defekt-Typen
  - Array out of bounds
  - Division by zero ...
- Hauptziel: semi-automatische Korrektur der Defekte





Kontakt: johanna.schneider@daimler.com