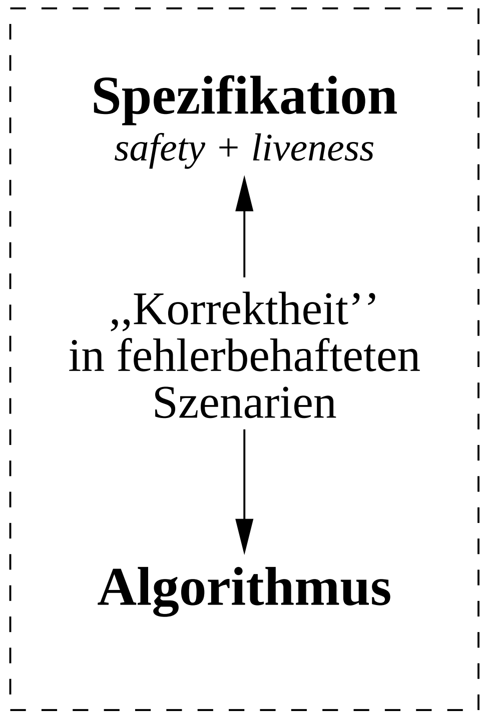
Fehlerbezogene Spezifikation

Felix Gärtner

TECHNISCHE UNIVERSITÄT DARMSTADT

Motivation und Hintergrund

- Notwendigkeit der Verifikation fehlertoleranter Algorithmen
- Probleme:
  - semi-formale Beweisführungen
  - uneinheitliche Fehlersemantiken
  - ad hoc Beweismethodiken



Szenario mit Fehlern

Spezifikation

safety + liveness

Korrektheit

Algorithmus

Szenario ohne Fehler

Vortragsgliederung

1. Einordnung des Themas und Motivation
2. der Begriff “fehlerbezogene Spezifikation”
3. Beispiel: *consensus*
4. erste Lösungsansätze

Fehlermodellierung

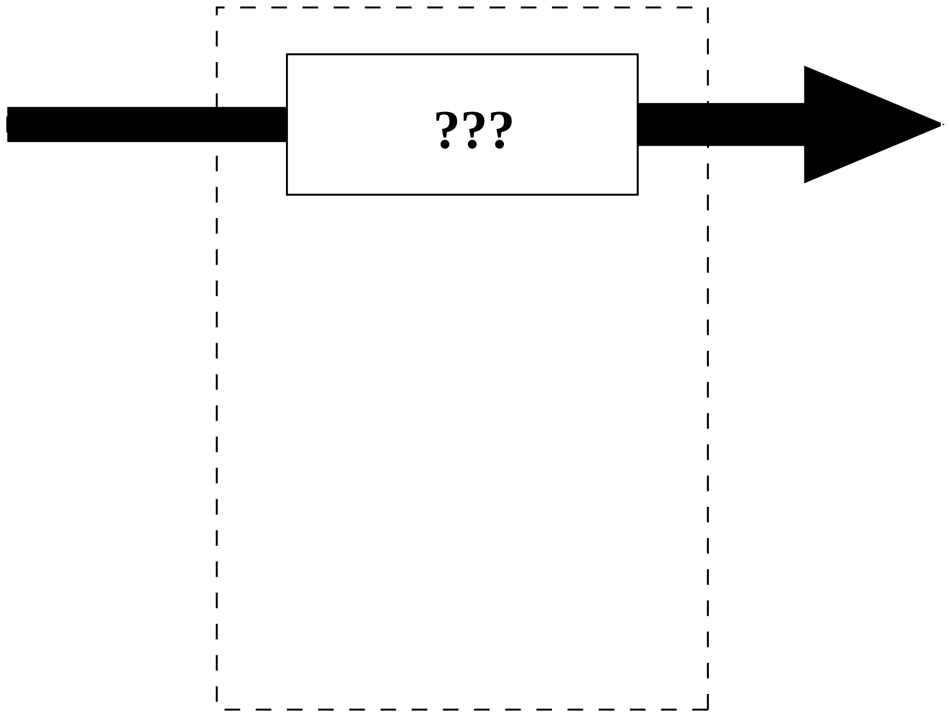
Motto: Man kann eine fehlerhafte Maschine nicht unterscheiden von einer Maschine, in die fehlerhaftes Verhalten "einprogrammiert" wurde.

Beispiel: *crash*-Fehlermodell

- Zusätzliche boolesche Variable *up* (anfangs *true*)
- *crash*-Aktion $up := false$
- Wenn $\neg up$, dann nichts mehr tun!

Fehlermodell ist Programmtransformation [Gärtner 1998]

Fehlermodell



Beispiel: *consensus*

- Netzwerk von Prozessen
- Operationen *propose*(v) und *decide*(v)
- Ein Algorithmus löst das *consensus*-Problem, gdw.
  1. allmählich jeder Prozeß *decide*(v) ausführt,
  2. alle Prozesse denselben Wert entscheiden, und
  3. der entschiedene Wert zuvor von mindestens einem Prozeß mittels *propose*(v) vorgeschlagen wurde.

(NB: Dies ist Spezifikation für den fehlerfreien Fall)

consensus-Spezifikation für *crash*

- Prozesse, die irgendwann abstürzen, heißen “fehlerhaft”.
- Nicht-fehlerhafte Prozesse heißen “korrekt”.
- Verfeinerte Spezifikation für *consensus*:
  1. Allmählich führt jeder *korrekte* Prozeß *decide*(v) aus.
  2. Alle korrekten Prozesse entscheiden denselben Wert.
  3. Der entschiedene Wert ist zuvor von mindestens einem korrekten Prozeß mittels *propose*(v) vorgeschlagen worden.

Alternative *consensus*-Spezifikation für *crash*

1. Allmählich führt jeder *korrekte* Prozeß *decide*(v) aus.
2. Alle Prozesse, die *decide*(v) ausführen, tun dies mit demselben Wert v.
3. Der entschiedene Wert wurde zuvor von mindestens einem Prozeß mittels *propose*(v) vorgeschlagen.

Kooperation von fehlerhaften Prozessen ist gewünscht und bei entsprechend gutartigem Fehlermodell auch möglich.

Spezifikationstransformation

- Zwei Extreme:
  - Restriktion des Problems auf alle korrekten Prozesse: Annahmen wie “Verbundenheit” müssen auch für korrekte Teilmenge gelten.
  - Spezifikation gilt weiterhin für alle Prozesse: Nur bei besonders gutartigem Fehlermodell möglich (z.B. “jede zweite Nachricht geht verloren”).
- Zwischenstufen beschreiben den Grad der Kooperation fehlerhafter Komponenten

Arbeitsplanung

Zu untersuchen sind:

- Abstrakte Formen der Fehlerspezifikation, verschiedene Abstraktionsniveaus.
- Beziehungen zwischen Fehlermodellen und dem “Grad der Kooperation”.
- Modularität von Spezifikationen und Fehlermodellen.

Zusammenfassung