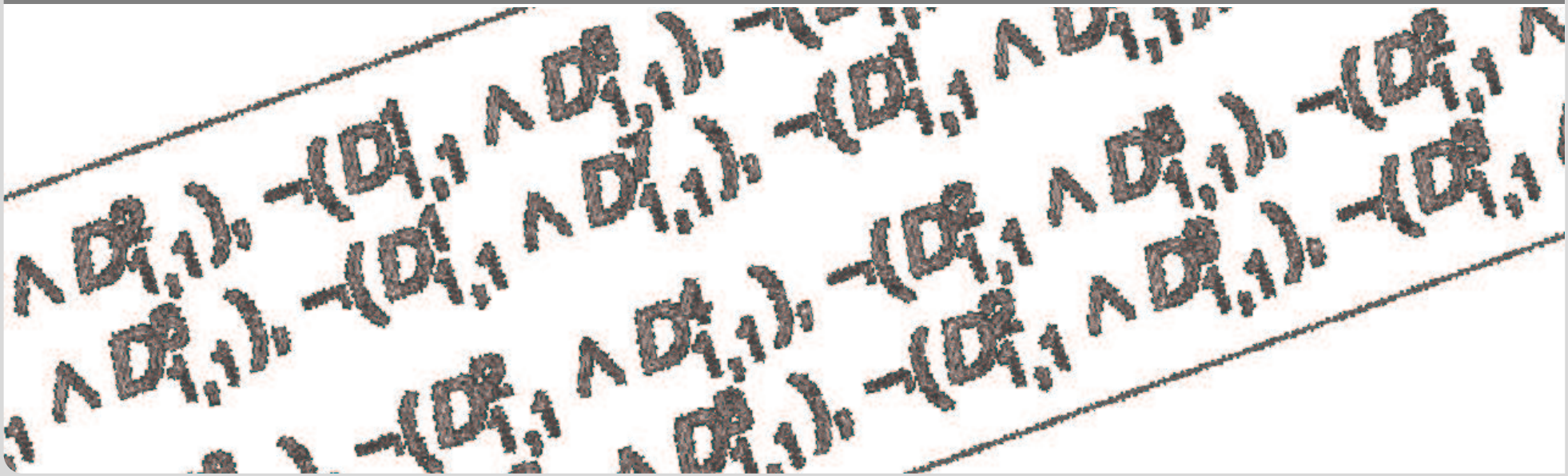
Karlsruher Institut für Technologie

Anwendungsorientierte Formale Verifikation
Praxis der Forschung im SoSe 2017

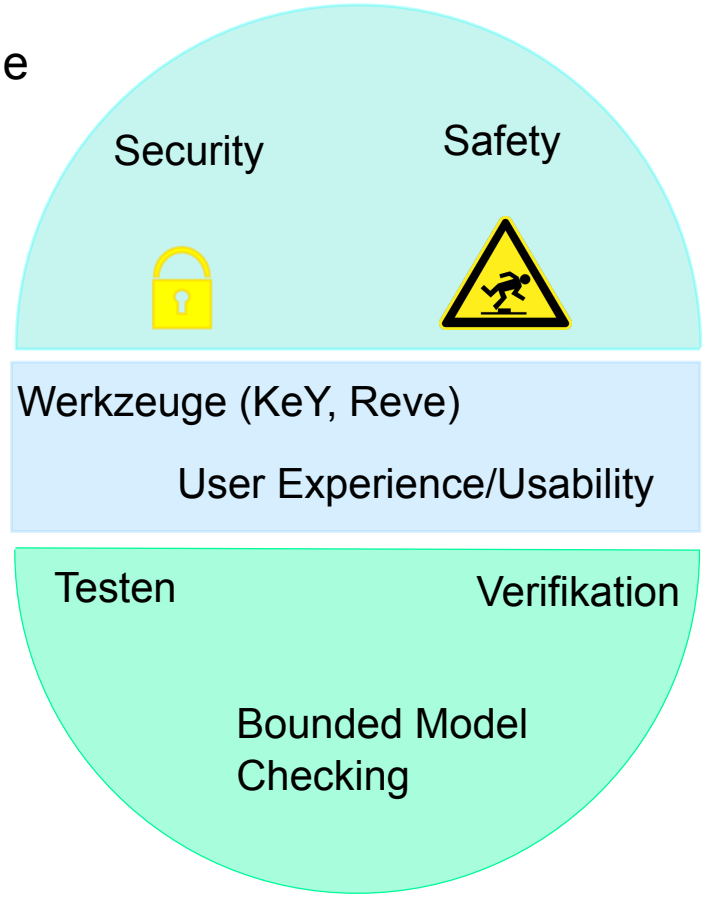
Bernhard Beckert

Institute for Theoretical Informatics, KIT



Themenbereiche der Gruppe

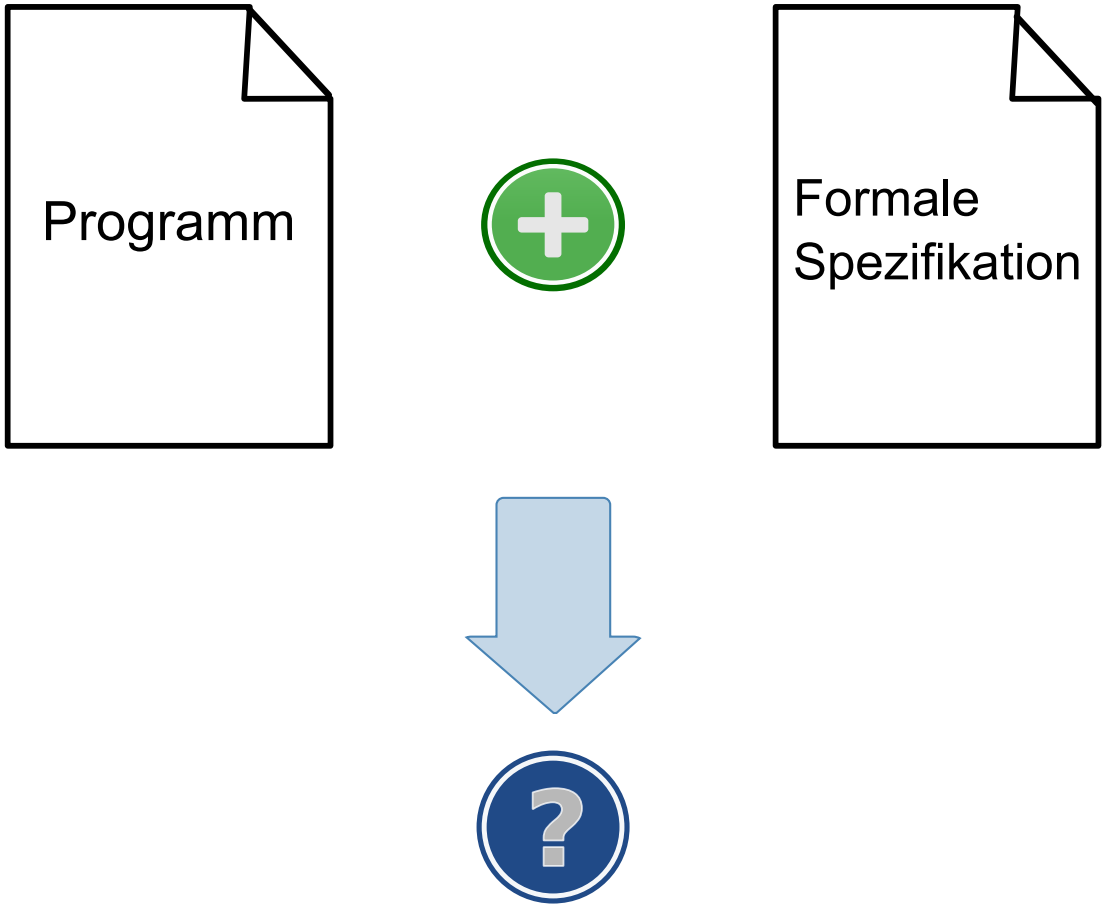
Anwendungsbereiche



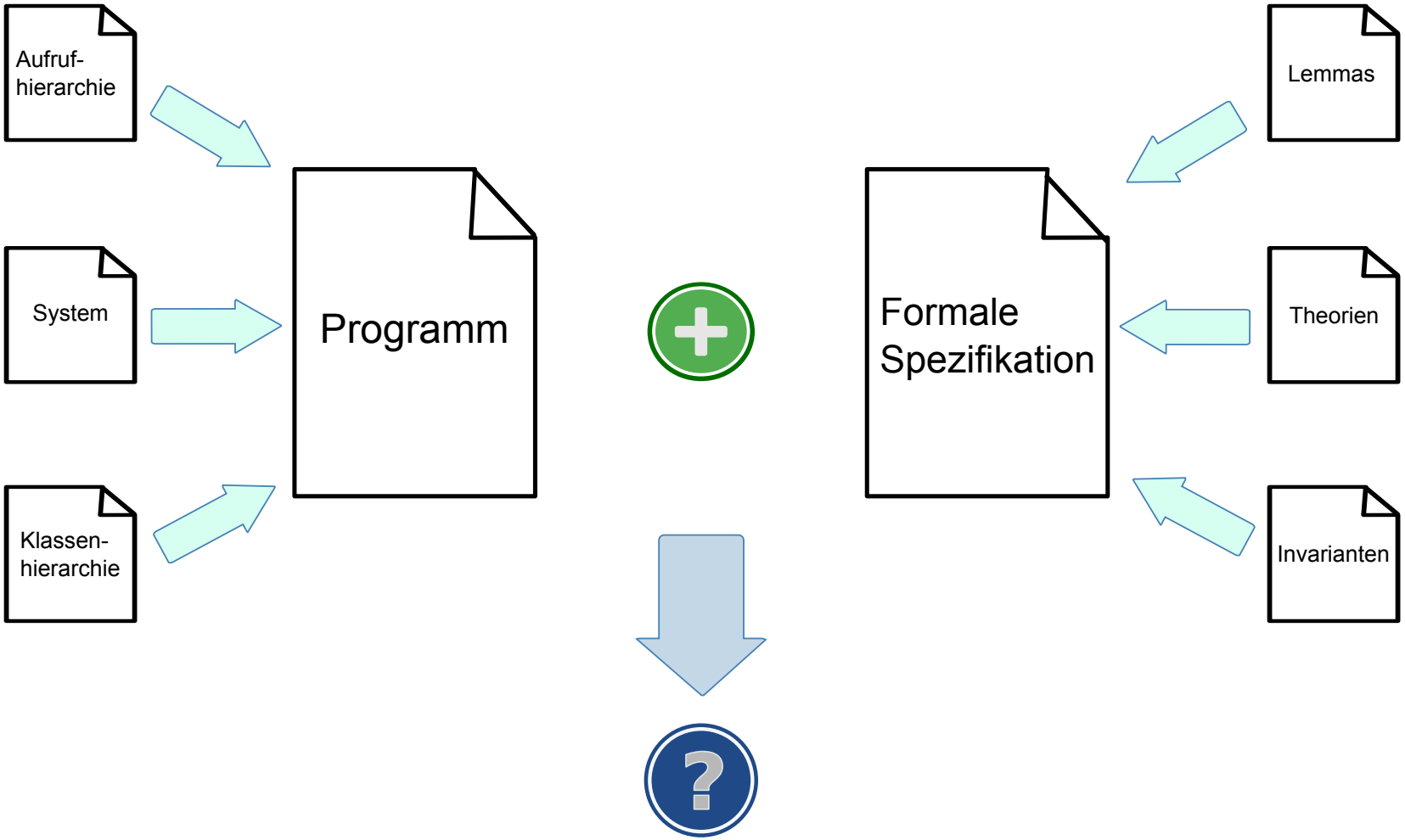
Methoden

Benutzbarkeit von Software-Verifikationssystemen

Programm Verifikation



Programm Verifikation



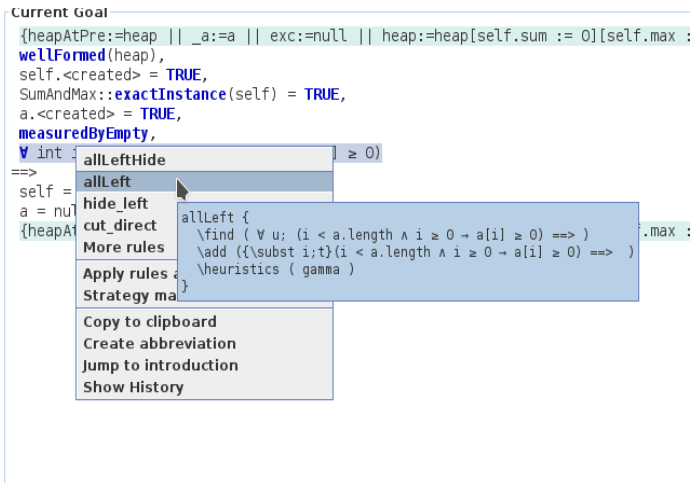
Sichten und Interaktionen

Programm-Ebene

```
method MultipleReturn(x: int, y: int) returns (more: int, less: int)
requires x > 0 && y > 0;
ensures  (y < x) ==> (less > 0) && (x <=y) ;
ensures more >= 0;
{
    more := x + y;
    less := x - y;
    assert more > less;
}
```

Logik-Ebene

```
script bool_simp_splits
 test := 6;
 foo:int;
 repeat
  foreach
    call bool_simp;
    cases
     case (test == 6) && (match '==> (?_ && ?_)@?P1'):
      andRight on ?P1;
     case match '(?_ || ?_)@?P2 ==>':
      orLeft on ?P2;
     case match '(?_ -> ?_)@?P3 ==>':
      impLeft on?P3;
     case match '==> ... bsum{?_}(?_,?N+1+1,?_)@P4 ... ':
      simp;
      bsum_def on ?P4;
      auto;
     case match '(f(c) = ?X)@P5 ==> ':
      applyEq @P5 on '==>... f(c) ...';
      auto;
     default:
      simp;
```



Sichten und Interaktionen

Programm-Ebene
(Annotationen im Code)

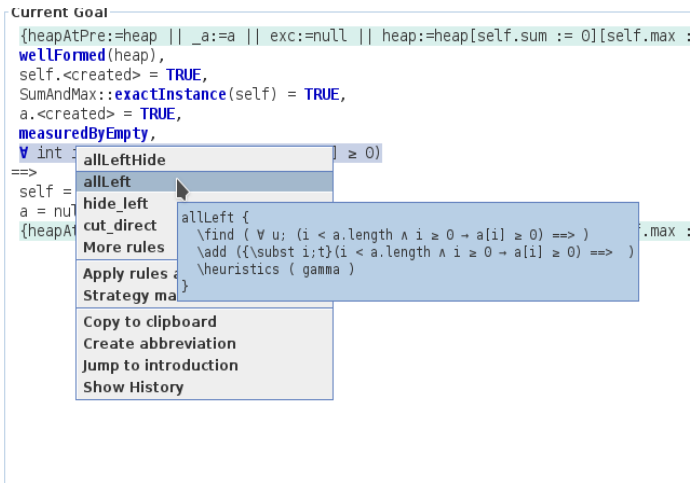
```
method MultipleReturn(x: int, y: int) returns (more: int, less: int)
requires x > 0 && y > 0;
ensures  (y < x) ==> (less > 0) && (x <=y) ;
ensures more >= 0;
{
    more := x + y;
    less := x - y;
    assert more > less;
}
```

Logik-Ebene

(Beweisskript)

```
script bool_simp_splits
 test := 6;
 foo:int;
 repeat
  foreach
    call bool_simp;
    cases
     case (test == 6) && (match '==> (?_ && ?_)@?P1'):
      andRight on ?P1;
     case match '(?_ || ?_)@?P2 ==>':
      orLeft on ?P2;
     case match '(?_ -> ?_)@?P3 ==>':
      impLeft on?P3;
     case match '==> ... bsum{?_}(?_,?N+1+1,?_)@P4 ... ':
      simp;
      bsum_def on ?P4;
      auto;
     case match '(f(c) = ?X)@P5 ==> ':
      applyEq @P5 on '==>... f(c) ...';
      auto;
     default:
      simp;
```

(Point & Click)



Sichten und Interaktionen

Programm-Ebene
(Annotationen im Code)

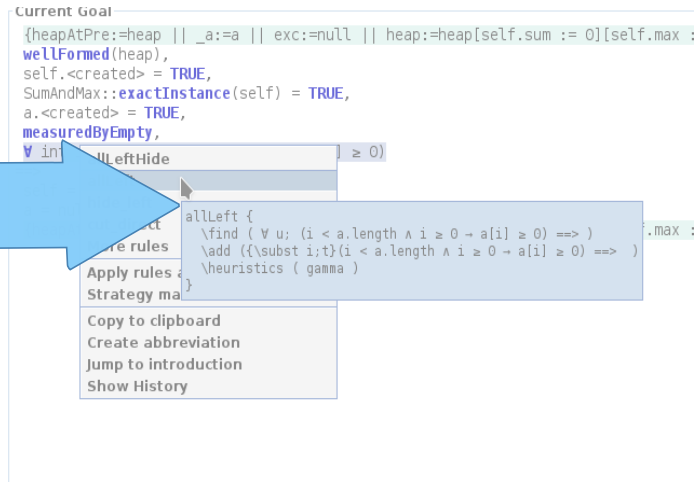
```
method MultipleReturn(x: int, y: int) returns (more: int, less: int)
requires x > 0 && y > 0;
ensures  (y < x) ==> (less > 0 && less <=y) ;
ensures more >= 0;
{
    more := x + y;
    less := x - y;
    assert more > less;
}
```

Logik-Ebene

(Beweisskript)

```
script bool_simp_splits
 test := 6;
 foo:int;
 repeat
  foreach
    call bool_simp;
    cases
     case (test == 6) && (match '==> (?_ && ?_)@?P1'):
      andRight on ?P1;
     case match '(?_ || ?_)@?P2 ==>':
      orLeft on ?P2;
     case match '(?_ -> ?_)@?P3 ==>':
      impLeft on?P3;
     case match '==> ... bsum{?_}(?_,?N+1+1,?_)@P4 ... ':
      simp;
      bsum_def on ?P4;
      auto;
     case match '(f(c) = ?X)@P5 ==> ':
      applyEq @P5 on '==>... f(c) ...';
      auto;
     default:
      simp;
```

(Point & Click)



Ziel

Verbesserung der Benutzbarkeit von Programm-Beweissystemen

Aufgaben

- Entwicklung eines integrierten Darstellungs-Konzepts basierend auf den verschiedenen Interaktionsparadigmen, bspw.:
  - Darstellung von Abhängigkeiten
  - Fließender Wechsel zwischen den Sichten
- Umsetzung des Konzepts in bestehendem Framework
- Evaluation durch Nutzerstudien (z.B. Eye-Tracking, Fragebogen, etc.)

Bei Interesse:

- Sarah Grebing `grebing@ira.uka.de` (ITI Beckert)
- Andrea Schankin `schankin@teco.edu` (TM Beigl)
- Mattias Ulbrich `ulbrich@kit.edu` (ITI Beckert)

Verifikation OO-Software für Produktionsanlagen

Verifikation objektorientierter Software für Produktionsanlagen

Karlsruher Institut für Technologie

Hintergrund

- Steigende Komplexität von Analagensteuerungen
- Einführung von Objekt-Orientiertierung

Regressionverifikation

Formale Methode zur Garantie der Verhaltensequivalenz

Methoden

Parserbau, Static Analysis, Model-Checking

Bei Interesse: Alexander Weigl (`weig@kit.edu`)

Verifikation objektorientierter Software für Produktionsanlagen

Karlsruher Institut für Technologie

Hintergrund

- Steigende Komplexität von Analagensteuerungen
- Einführung von Objekt-Orientiertierung

Regressionverifikation

Formale Methode zur Garantie der Verhaltensequivalenz

Methoden

Parserbau, Static Analysis, Model-Checking

Bei Interesse: Alexander Weigl (`weig@kit.edu`)

Verifikation objektorientierter Software für Produktionsanlagen

Karlsruher Institut für Technologie

Hintergrund

- Steigende Komplexität von Analagensteuerungen
- Einführung von Objekt-Orientiertierung

Regressionverifikation

Formale Methode zur Garantie der Verhaltensequivalenz

Methoden

Parserbau, Static Analysis, Model-Checking

Bei Interesse: Alexander Weigl (`weig@kit.edu`)

Verifikation objektorientierter Software für Produktionsanlagen

Karlsruher Institut für Technologie

Hintergrund

- Steigende Komplexität von Analagensteuerungen
- Einführung von Objekt-Orientiertierung

Regressionverifikation

Formale Methode zur Garantie der Verhaltensequivalenz

Methoden

Parserbau, Static Analysis, Model-Checking

Bei Interesse: Alexander Weigl (`weig@kit.edu`)