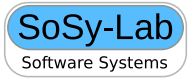
Formale Spezifikation und Verifikation

Wintersemester 2024

Prof. Dr. Gidon Ernst
gidon.ernst@lmu.de

Software and Computational Systems Lab
Ludwig-Maximilians-Universität München, Germany

September 30, 2024



Spezifikation und Testen eingebetter Systeme

Plan

Bisher: Lineare Temporale Logik

- Syntax und Semantik von Formeln, diskrete Zeitschritte
- Sicherheits- und Lebendigkeitseigenschaften
- Model Checking mit Büchi-Automaten

Heute: Eingebettete Systeme

- Controller + Modell der physikalischen Umgebung:
  $\rightarrow$ kontinuierliche Zeit und Werte
- Modellierung und Spezifikation mit Hybriden Automaten und MTL/STL
- Gezieltes Aufdecken gefährlichen Verhaltens durch *Simulation*

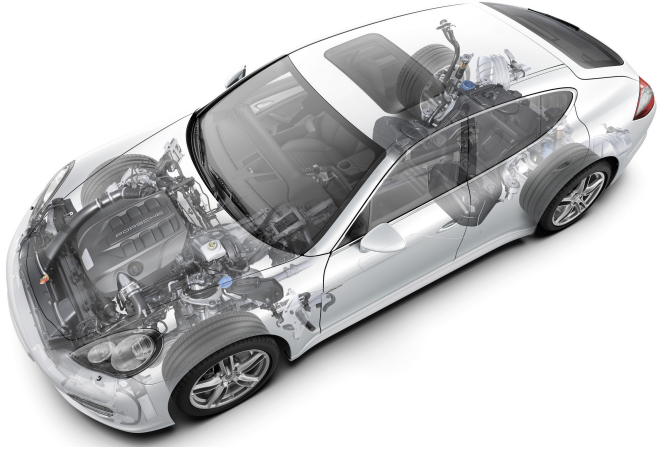
Nächste Woche: aus der Praxis, bis ca 12:15

Reaktive Systeme

```
int sqliteProcessJoin(Parse *pParse, Select *p){
ist *pSrc;                    /* All tables in the FROM clause *
, j;                          /* Loop counters */
t SrcList_item *pLeft;        /* Left table being joined */
t SrcList_item *pRight;       /* Right table being joined */

= p->pSrc;
 = &pSrc->a[0];
ht = &pLeft[1];
=0; i<pSrc->nSrc-1; i++, pRight++, pLeft++){
ble *pRightTab = pRight->pTab;
 isOuter;

 NEVER(pLeft->pTab==0 || pRightTab==0) ) continue;
Outer = (pRight->fg.jointype & JT_OUTER)!=0;

When the NATURAL keyword is present, add WHERE clause terms f
every column that the two tables have in common.

 pRight->fg.jointype & JT_NATURAL ){
if( pRight->pOn || pRight->pUsing ){
 sqlite3ErrorMsg(pParse, "a NATURAL join may not have "
    "an ON or USING clause", 0);
 return 1;
}
for(j=0; j<pRightTab->nCol; j++){
 char *zName;   /* Name of column in the right table */
 int iLeft;     /* Matching left table */
 int iLeftCol;  /* Matching column in the left table */

 zName = pRightTab->aCol[j].zName;
 if( tableAndColumnIndex(pSrc, i+1, zName, &iLeft, &iLeftCol)
```

Eingebettete Systeme

"Cyber-Physical Systems"



Eingebettete Systeme—Charakteristika

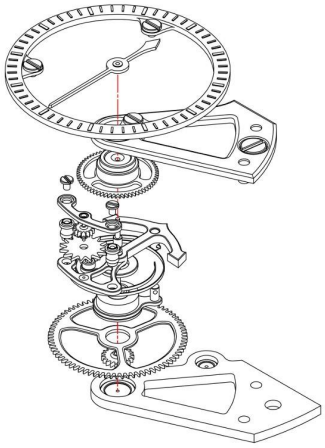
"Cyber-Physical Systems"

- Interaktion: technisches System $\leftrightarrow$ Umwelt
- zeitliches Verhalten
- Modellierung: hybride Automaten, Matlab, ...
- Spezifikation *Quantitative* (lineare) Temporallogik
  (Metric Temporal Logic (MTL), Signal Temporal Logic (STL))

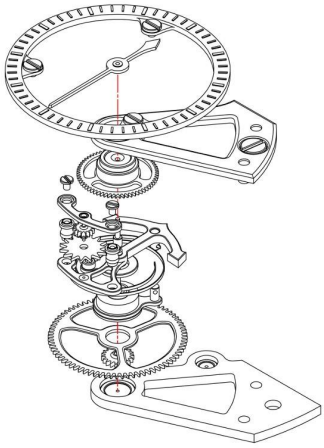
Physikalische Systeme: Außensicht

- Eingabesignal $x \in T \to \mathbb{R}^n$
- Ausgabesignal $y \in T \to \mathbb{R}^m$

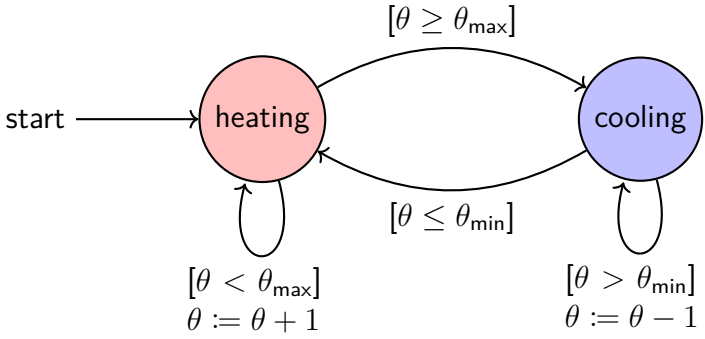
Beispiel Uhr: Außensicht



Beispiel Uhr: Modell



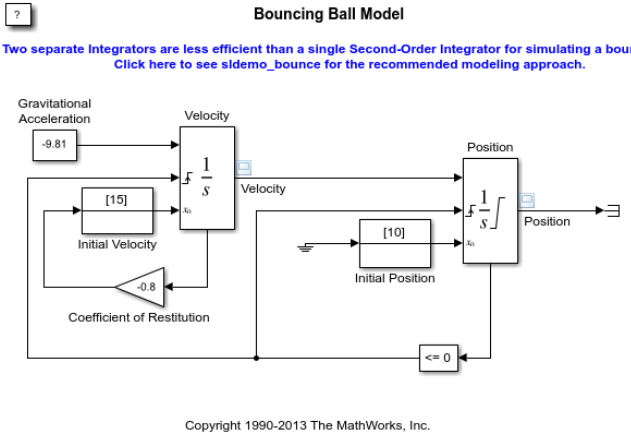
Reminder: Thermostat diskret Modelliert

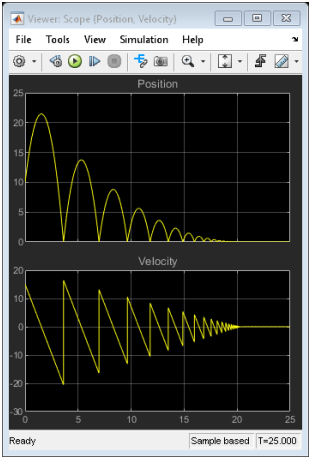


Modellierung: Automat mit Variablen ($\approx$ Kontrollflussautomat)

Beispiel: Thermostat als hybrider Automat

Beispiel: Bouncing Ball





Trajektorien des Bouncing Balls

Bouncing Ball als hybrider Automat

Bouncing Ball als hybrides Programm

```
assume H >= 0 && g > 0; // Starthöhe und Gravitation
assume 0 < c < 1;       // Dämpfung beim Aufprall

x := H, v := 0;

loop {
    continuously {
        x' == v, v'== -g
    }
    invariant (x >= 0)
    until     (x == 0);

    v := -c * v;
}
```

Modellierung physikalischer Systeme

- Eingabesignal $x \in T \to \mathbb{R}^n$
- Ausgabesignal $y \in T \to \mathbb{R}^m$
- Differenzialgleichungen: $\delta y / \delta t = f(x)$
- Diskrete (Betriebs-)modi

- Kontinuierliches Verhalten
- Diskrete Transitionen (“Sprünge”)
- Abstraktion vs Präzision bei physikalischen Phänomenen

Problem: analytische Ansätze (Lösen von Gleichungen) oft nicht möglich

Ansätze:

- eingeschränkte Modelle, z.B. gezeitete Automaten
- Testen/Simulation
- interaktive Beweissysteme, z.B. KeYmaeraX ($\to$ Marvin Brieger)

Simulation der Trajektorie des Bouncing Balls

Gleichungen: $dv/dt = -g$ und $dx/dt = v$
diskreter Übergang bei $x = 0$: $v \rightsquigarrow -v$;

Schrittweise Integration mit `dt` *klein genug* (z.B. `0.01`); z.B. mit dem Euler-Verfahren (besser: Runge-Kutta)

```
class BouncingBall {
    double t,v,x;

    void next(double dt) {
        t += t;
        v += dt * -Constants.g;
        x += dt * v;

        if(x <= 0)
            v = -v;
    }
}
```

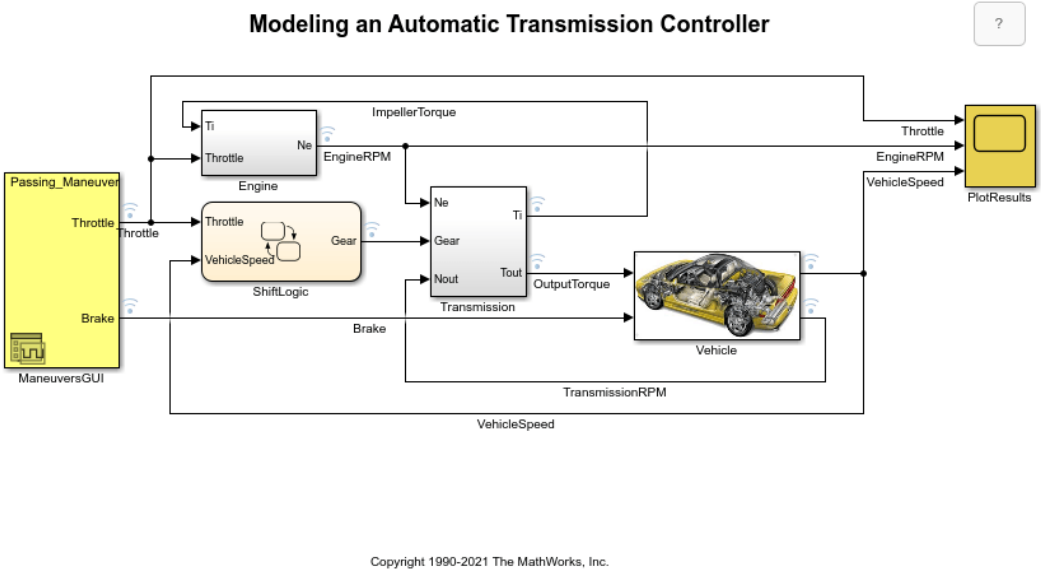
Spezifikation: Metric/Signal Temporal Logic

Wie LTL, aber kontinuierliche Zeit (“Signale” statt Abläufe)

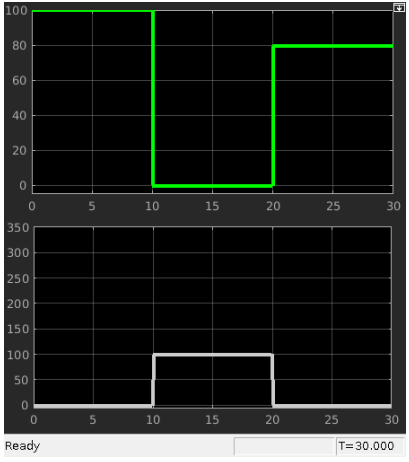
- temporale Operatoren mit Zeitintervallen:
  - $\Box_{[a,b]}\phi$ gilt zum Zeitpunkt t wenn ϕ ab allen Zeitpunkten $t+a \ldots t+b$ gilt
  - $\Diamond_{[a,b]}\phi$ gilt zum Zeitpunkt t wenn ϕ ab irgendwann zwischen $t+a \ldots t+b$ gilt
  - $\phi \, \mathcal{U}_{[a,b]} \, \psi$ analog
  - kein “next” $\circ\phi$,

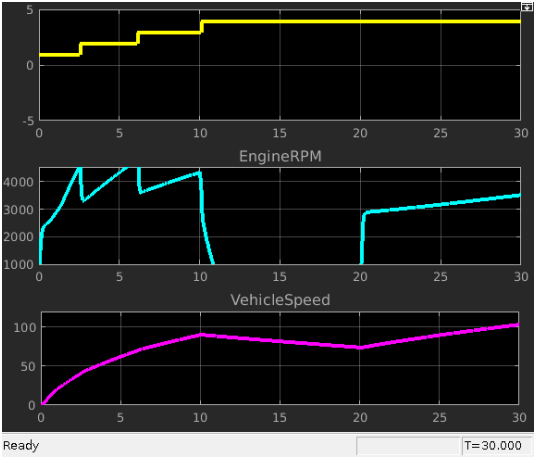
Beispiele (Bouncing Ball)

Beispiel: Automatikgetriebe



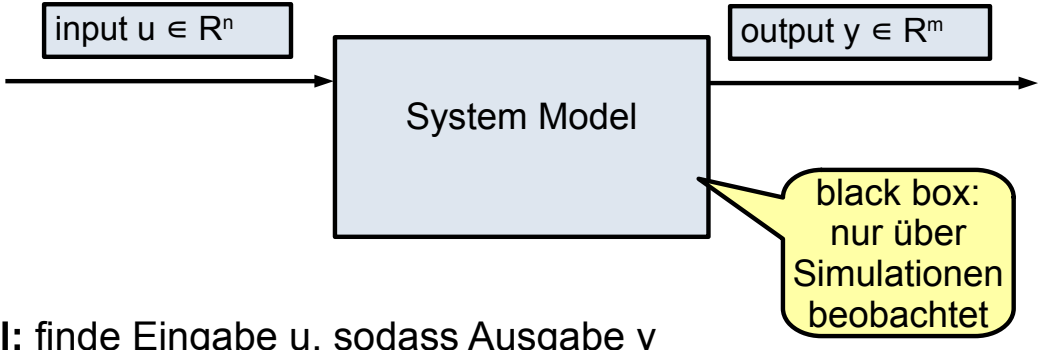
Beispiel: Automatikgetriebe





Eigenschaften: Automatikgetriebe

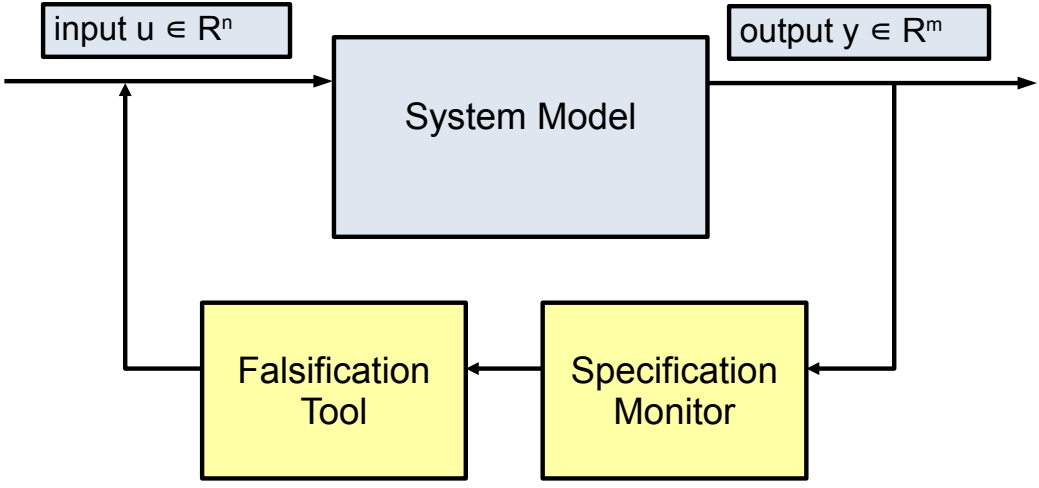
Falsifikation



Ziel: finde Eingabe u, sodass Ausgabe y
eine gegebene temporallogische Formel verletzt

"wie kann ich ein Auto gegen die Wand fahren"

Falsifikation



Naiver Ansatz: Zufallstesten

Beispiel: Verletzung für $\square(v < 120)$ finden

Naiver Ansatz: Zufallstesten

Beispiel: Verletzung für $\square(v < 120)$ finden

z.B. analog zu Property-based:

```
@Property
void test(Model model, Formula phi,
          @ForAll double[] throttle, @ForAll double[] brake)
{
        Trace trace = model.simulate(throttle, brake);

        if(!phi.satisfiedOn(trace)) {
                throw RequirementViolated(...);
        }
}
```

Metric Temporal Logic: “Robustheit”

Gradientenverfahren

Beispiel: Verletzung für $\square(v < 120)$ finden

Stochastische Optimierung [Nelder/Mead, CMA-ES]

Beispiel: Verletzung für $\square(v < 120)$ finden

Inkrementelle Optimierung

Beispiel: Verletzung für $\square(v < 120)$ finden

Diskrete Baumsuche [RRT, aLVTS]

Beispiel: Verletzung für $\square(v < 120)$ finden

ARCH-COMP

Wettbewerb seit 2014:

- Ziel: Vergleich von Analysemethoden für hybride Systeme
- unterschiedliche Kategorien:
  - rein kontinuierlich vs diskrete Übergänge
  - lineare vs nichtlineare Differenzialgleichung
  - Verifikation vs Testen

ARCH-COMP

Wettbewerb seit 2014:

- Ziel: Vergleich von Analysemethoden für hybride Systeme
- unterschiedliche Kategorien:
  - rein kontinuierlich vs diskrete Übergänge
  - lineare vs nichtlineare Differenzialgleichung
  - Verifikation vs Testen

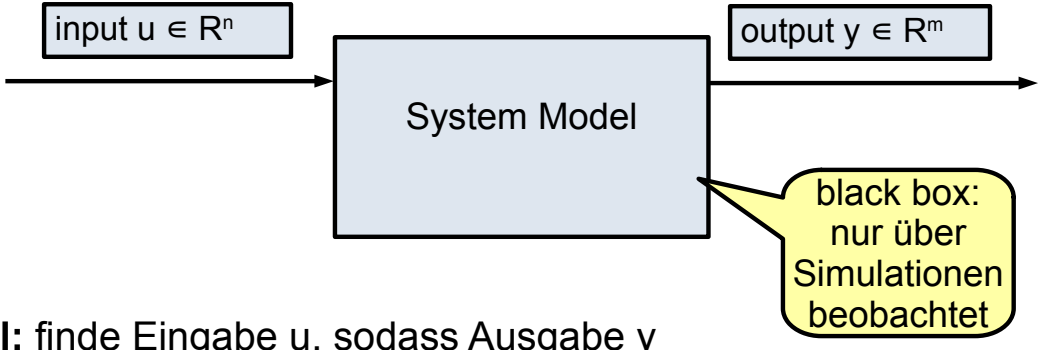
Impact

- Schnittstelle zwischen Industrie und Forschung
- Etablierung von Benchmarks $\rightarrow$ Vergleichbarkeit, Reproduzierbarkeit
- Übertrag der Ansätze auf Neuronale Netze

ARCH-COMP: Falsifikation

https://cps-vo.org/group/ARCH/FriendlyCompetition

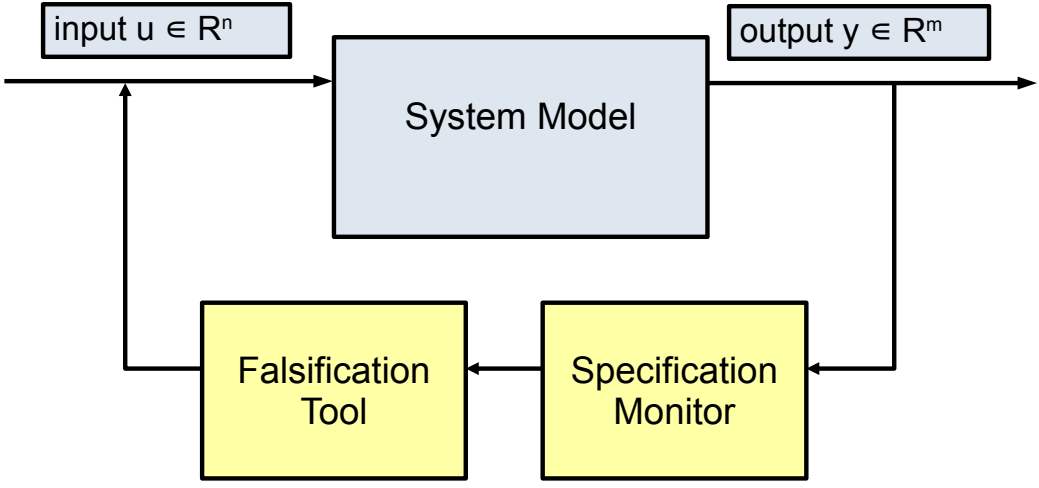
Falsifikation



Ziel: finde Eingabe u, sodass Ausgabe y
eine gegebene temporallogische Formel verletzt

"wie kann ich ein Auto gegen die Wand fahren"

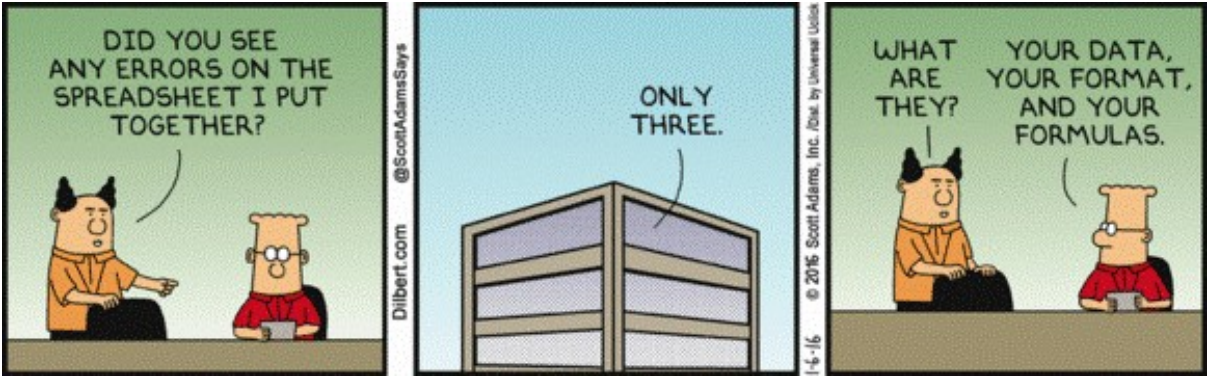
Falsifikation



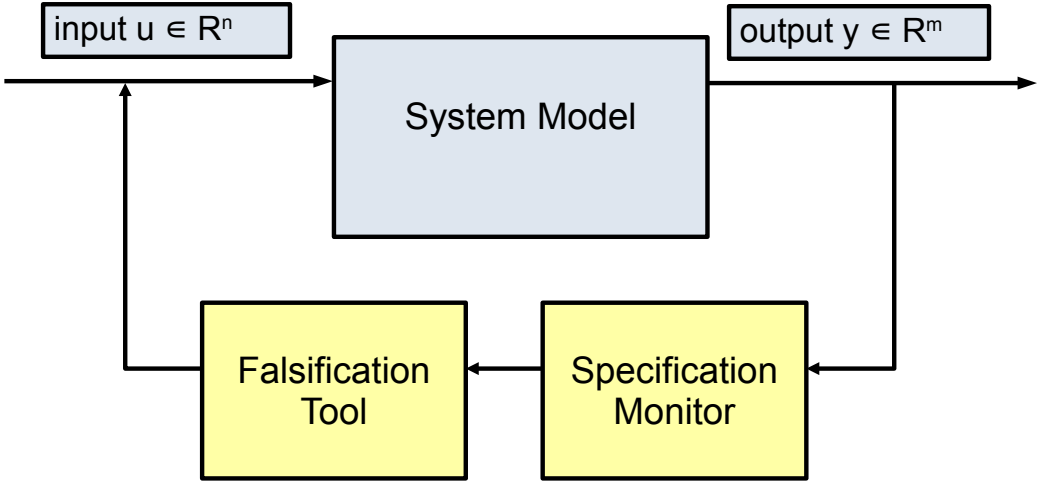
Organisation

- Benchmarks:
  - 7 Modelle
  - ca 25 Spezifikationen insgesamt
- Abfolge
  - festlegen der Benchmarks
  - TeilnehmerInnen führen Experimente durch
  - Ergebnisse werden gesammelt
  - Bericht wird geschrieben

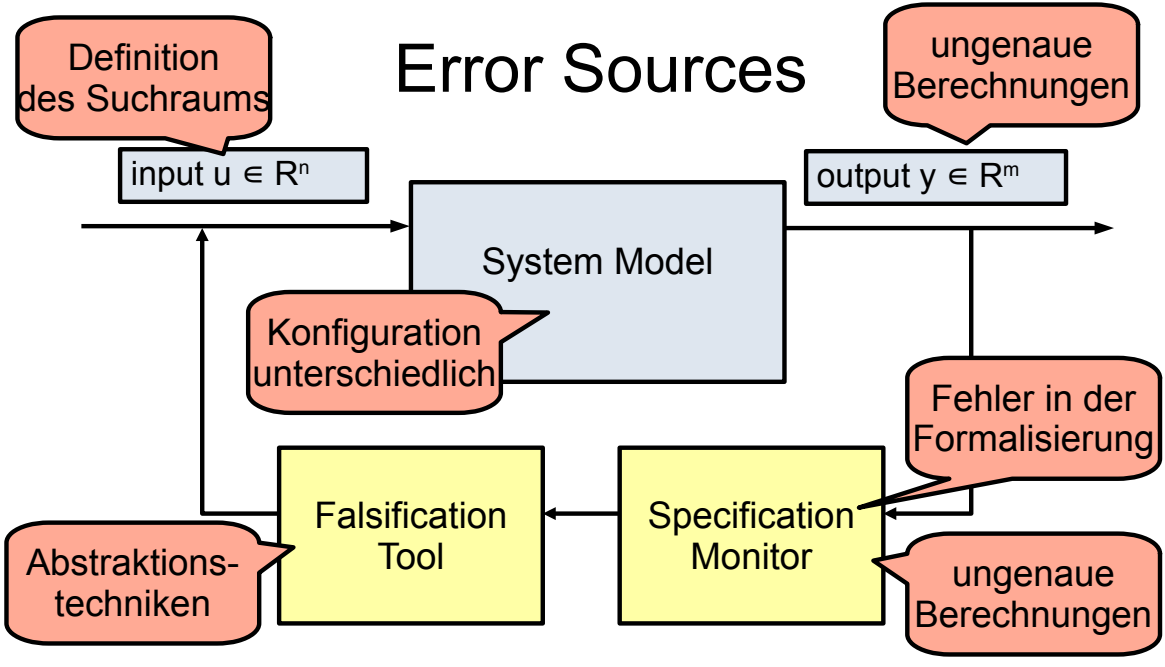
Validierung



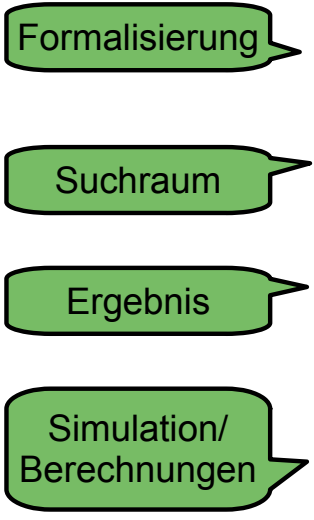
Fehlerquellen



Error Sources



2021/FALS/Validation.md




```
property: AT1
formula: □_[0, 20] (speed < 120)
input is within bounds
using stop time as provided: 20.0
falsified is correct
expected robustness -0.014
computed robustness -0.013
robustness error 0.001
[...]
```

Validierung über die Robustheit

um wie viel haben wir die Wand verfehlt?

reported (by participants)	computed (by validator)	confirmed falsification?
r < 0	r < 0	
r < 0	r < threshold	
r >= 0	_	

Erwin Wurm: Truck, 2015

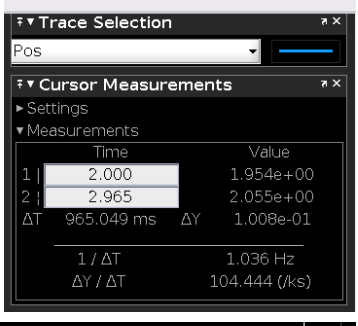
Ergebnisse

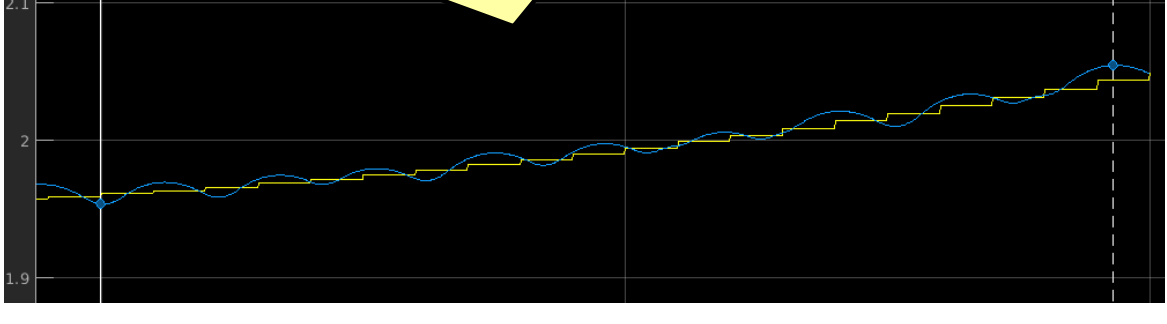
- Viele Diskrepanzen entdeckt
  - Dokumentation: ungenau, unvollständig, inkonsistent
  - vereinzelte Fehler: Formalisierung, Suchraum
  - Format der berichteten Ein-/Ausgabesignale

- Zusammenfassung
  - viele vorige Ergebnisse bestätigt | teilweise falsche Ergebnisse
  - die größten und offensichtlichsten Fehler sind beseitigt
  - einzelne subtilere Fehler bleiben (unklar warum)

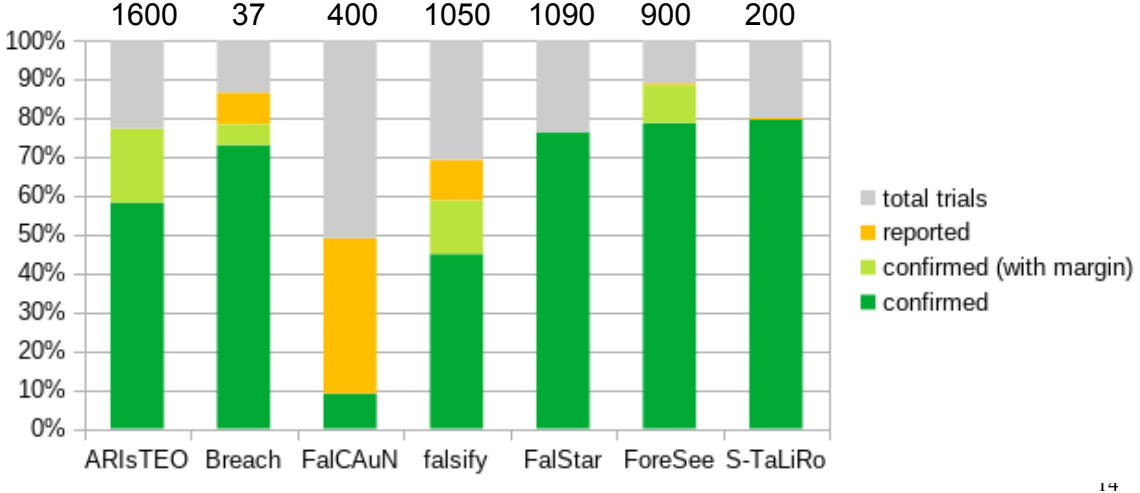
Fehlersuche

manuelle Überprüfung:
verletzt das Signal die Spec?





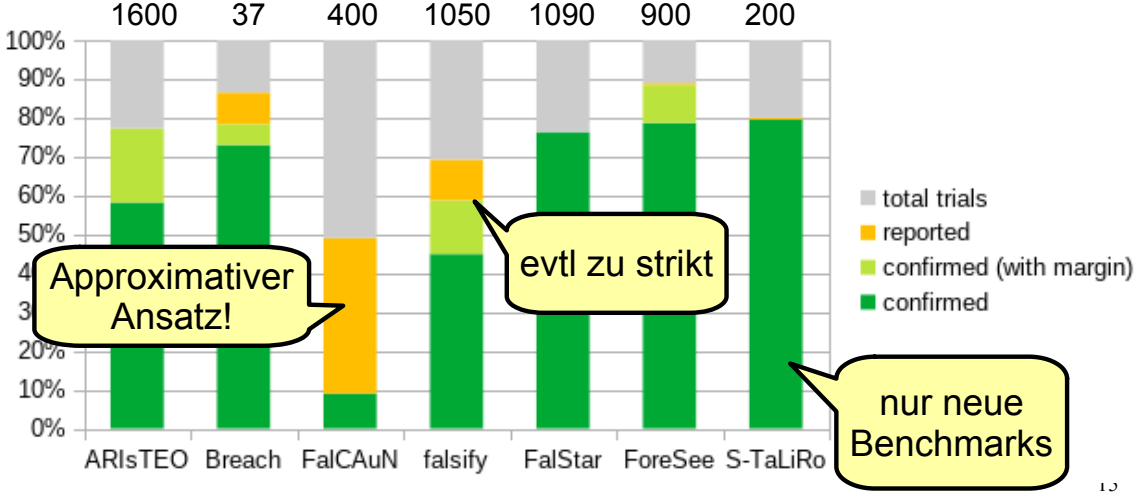
“Ergebnisse”


1600
37
400
1050
1090
900
200
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
ARIsTEO
Breach
FalCAuN
falsify
FalStar
ForeSee
S-TaLiRo
total trials
reported
confirmed (with margin)
confirmed

“Ergebnisse”

deterministisch (+ eine falsche Spec)

brauchen mehr Validatoren!



Approximativer Ansatz!

evtl zu strikt

nur neue Benchmarks

License