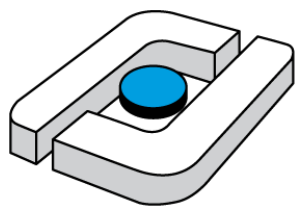
Verifikation von UML-Statecharts unter besonderer Berücksichtigung von Speicherverbrauch und Laufzeit des Model Checkers

Dipl.-Inform. Christian Ammann

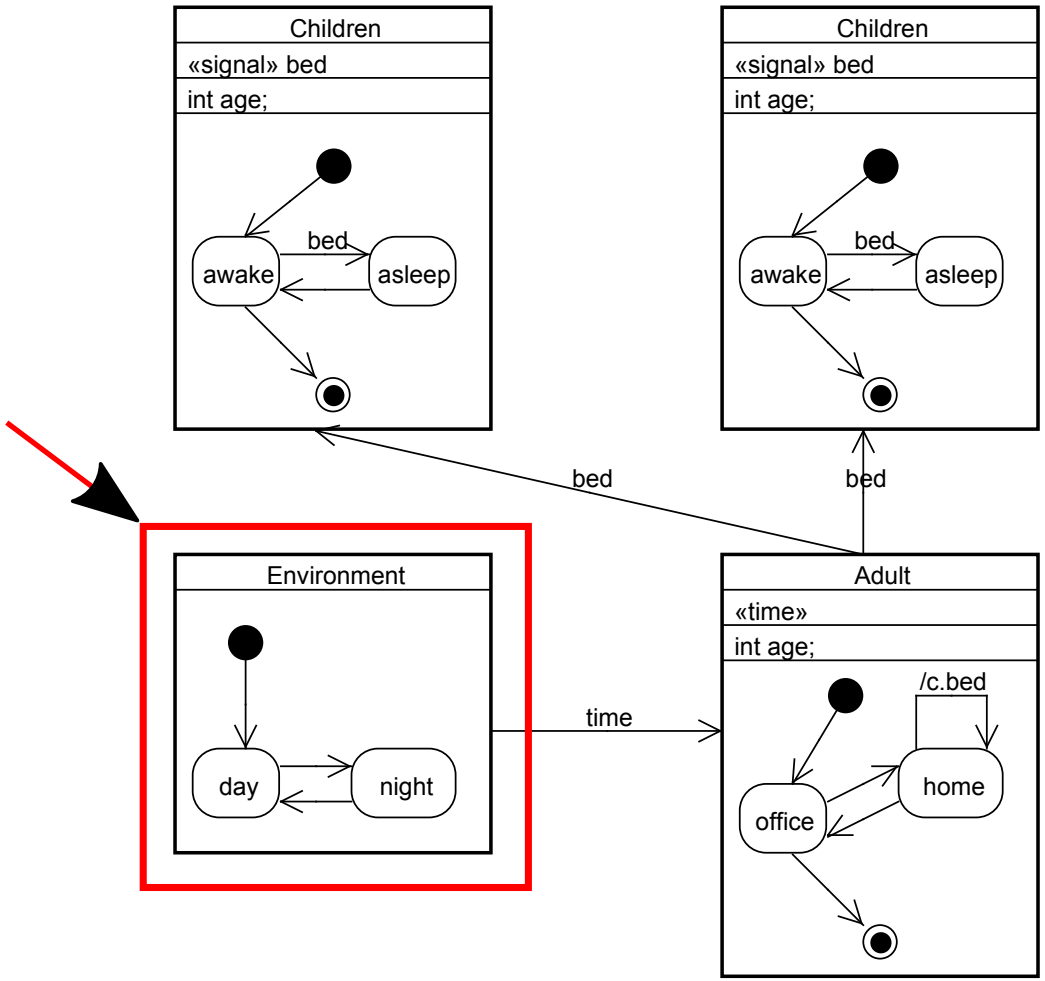
Hochschule Osnabrück
Fakultät für Ingenieurwissenschaften und Informatik

30.6.2011

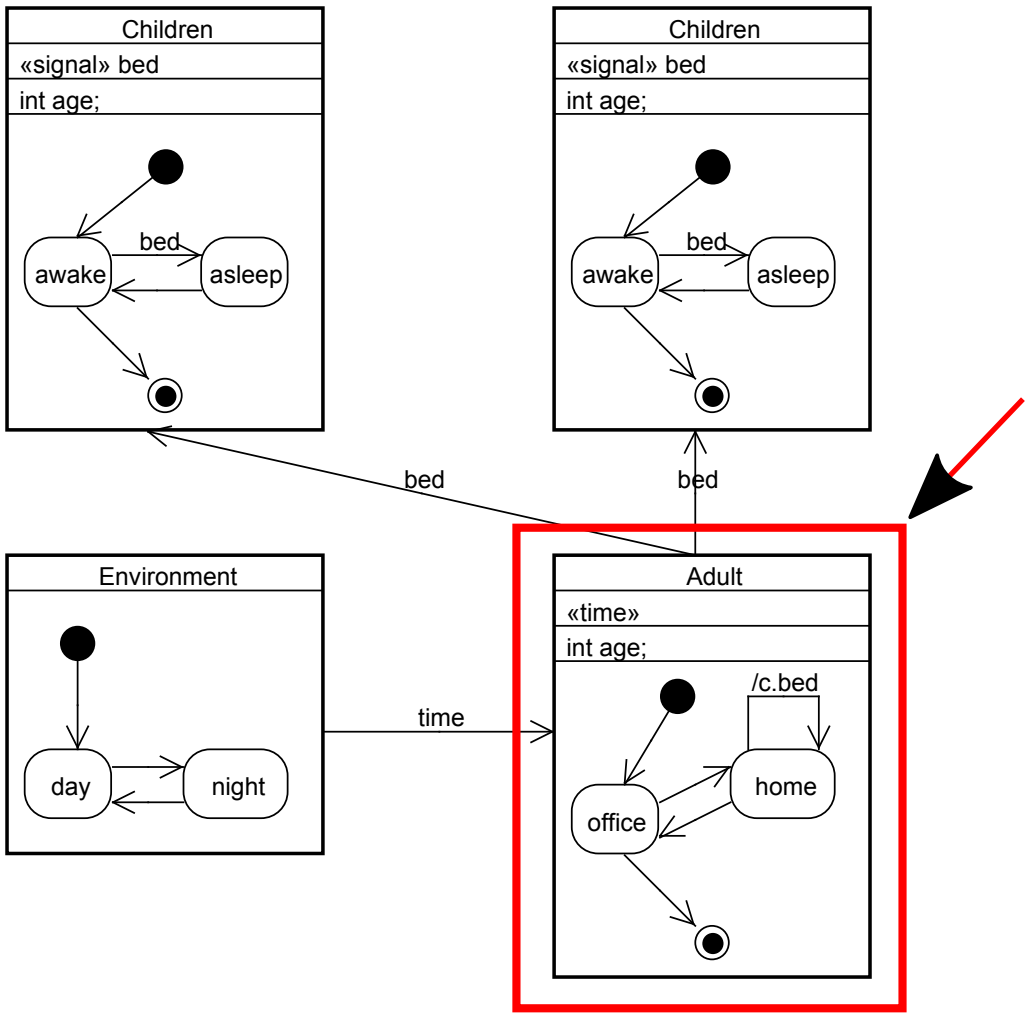


Inhalt

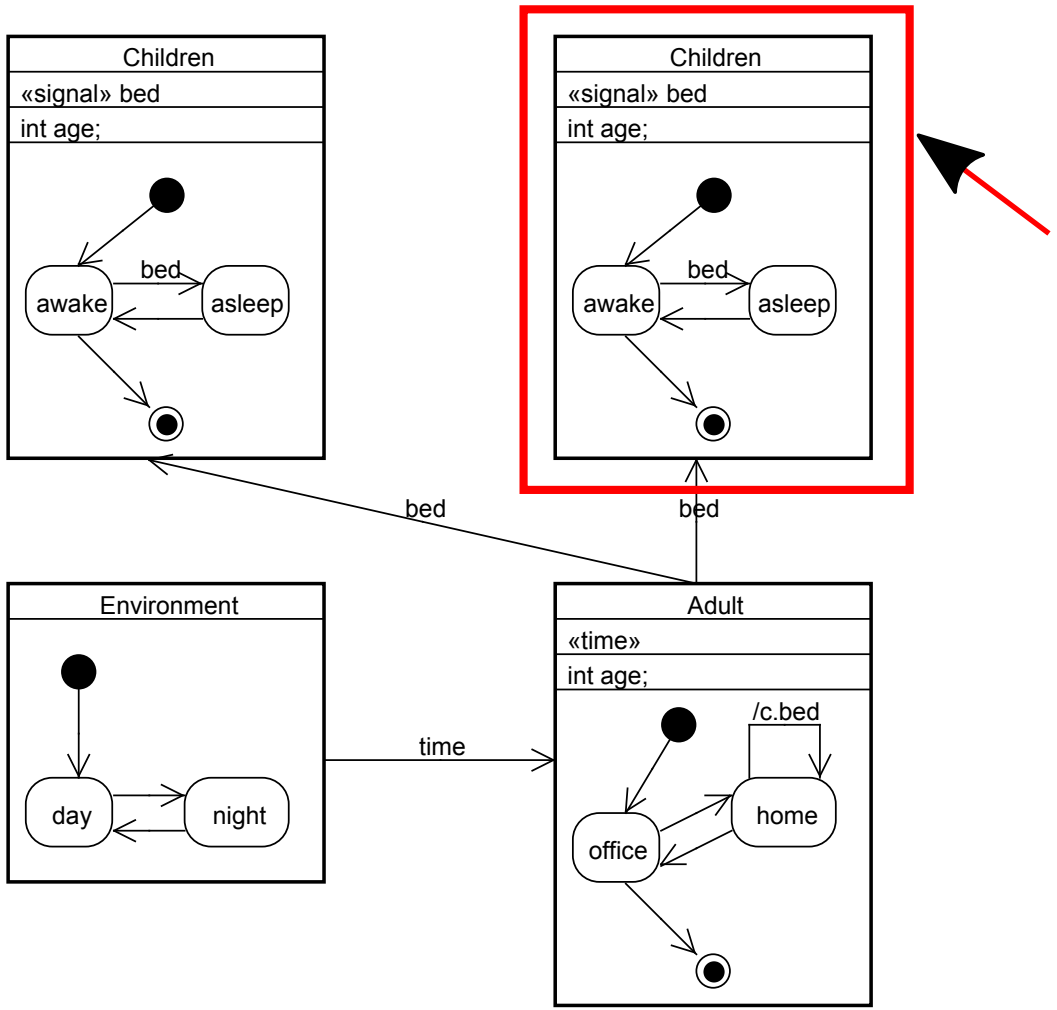
Motivation: Environment-Objekt



Motivation: Adult-Objekt



Motivation: Children-Objekt



Zustandsraum

Zustand des Gesamtsystems ist:

- Aktueller Zustand Environment
- Aktueller Zustand Adult
- Variablenbelegung von Adult
- Aktuelle Zustände der Children-Objekte
- Variablenbelegungen der Children-Objekte

UML Statecharts

- Hierarchie
- Nebenläufigkeit
- synchrone und asynchrone Events
- Actions
- Transitionen mit Conditions
- Semantik nicht vollständig definiert

Spin

- Spin steht für **Simple Promela Interpreter**
- Promela steht für **Protocol/Process Meta Language**
- Beschreiben von Modellen und Anfragen
- Dynamisches Erstellen nebenläufiger Prozesse
- Synchrone/Asynchrone Kommunikation

Domänenspezifische Sprache UDL

- Entry und Exit Actions
- Signal- und Call-Events
- Transition mit Trigger, Guard oder Effect
- Nebenläufigkeit
- Implementiert mit Xtext
- Automatische Übersetzung nach Promela mit Xpand

Optimierungen in UDL und Promela

- Symmetrie
- Statement-Merging
- Zurücksetzen von Hilfsvariablen

Statechart-Beispiel in UDL

example.udl

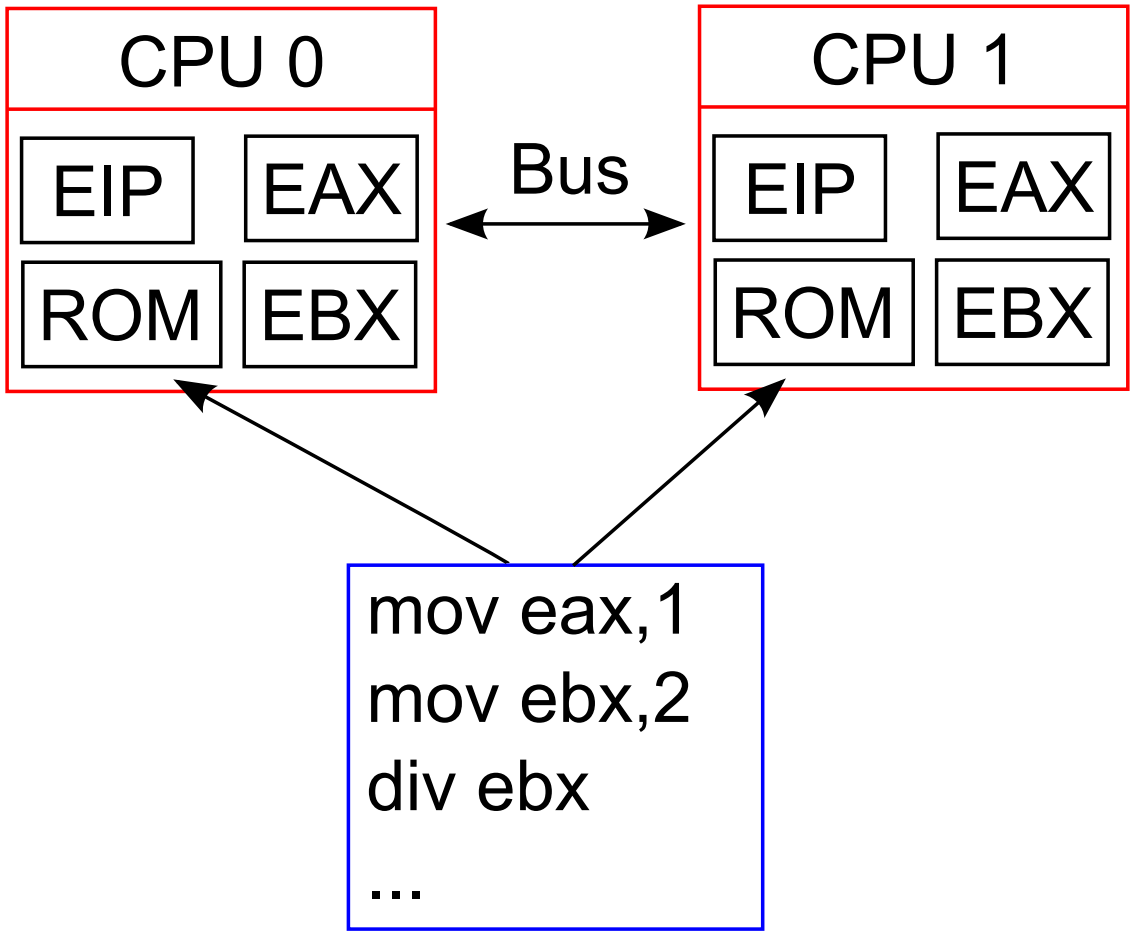
```
class StateMachine{
    byte a = 0;
    signal exec {byte b}

    initialstate start{
        -> calculate{
            trigger{exec}
        }
    }

    simplestate calculate{
        entry{a = a * exec.b;}
        -> stop{}
    }

    finalstate stop{}
}
```

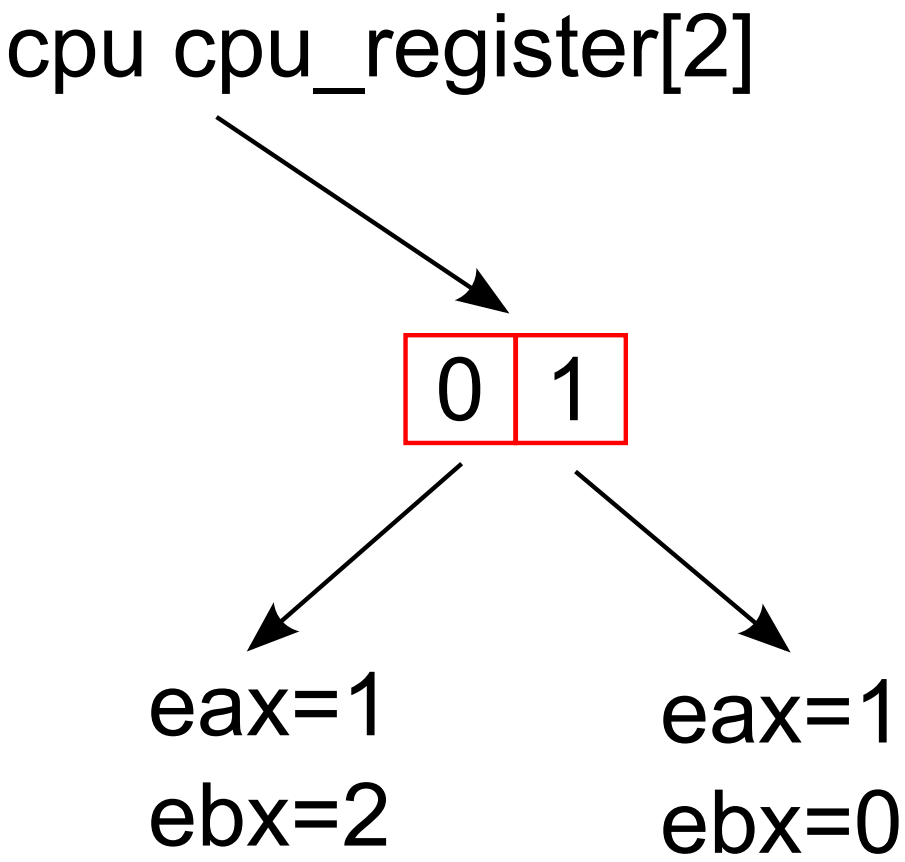
Symmetrie


CPU 0
EIP
EAX
ROM
EBX
Bus
CPU 1
EIP
EAX
ROM
EBX
mov eax,1
mov ebx,2
div ebx
...

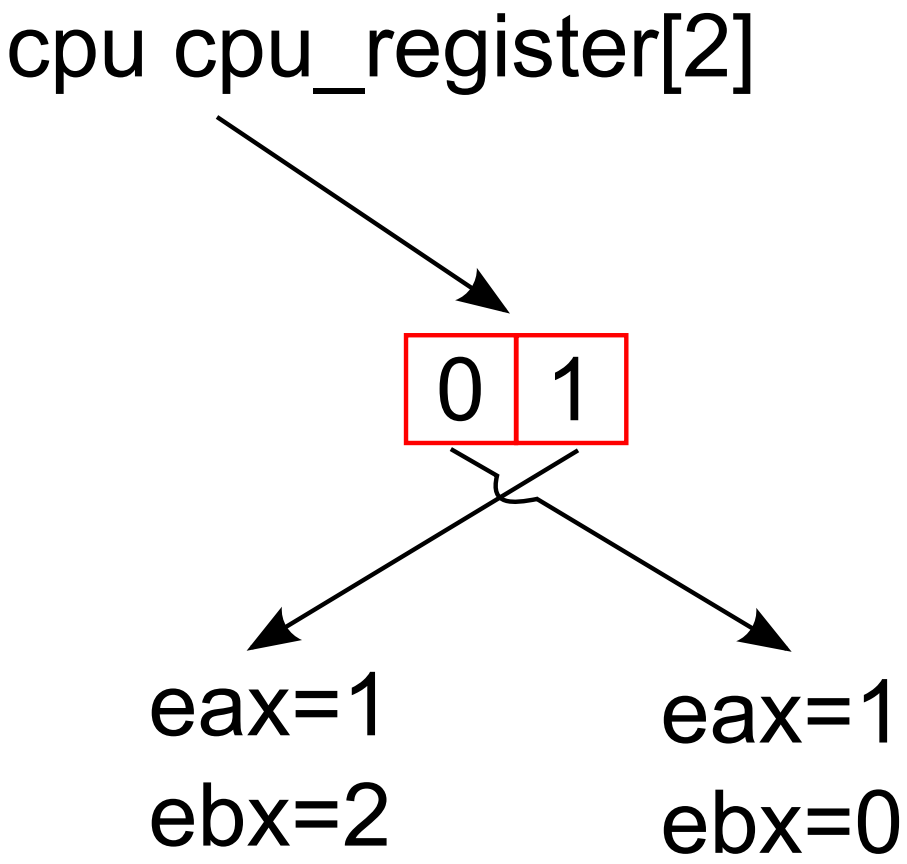
Symmetrie

```
1  typedef cpu{
2      int eax=0;
3      int ebx=0;
4  }
5
6  cpu cpu_register[2];
7
8  proctype rom(){
9     cpu_register[_pid].eax=1
10    ...
11 }
```

Symmetrie

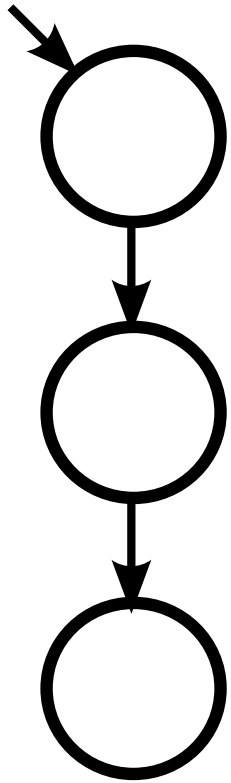


Symmetrie



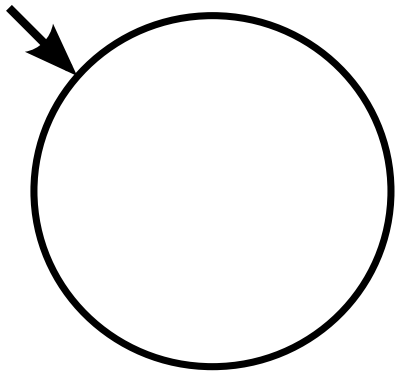
Statement Merging

Promela	SPIN Verifier

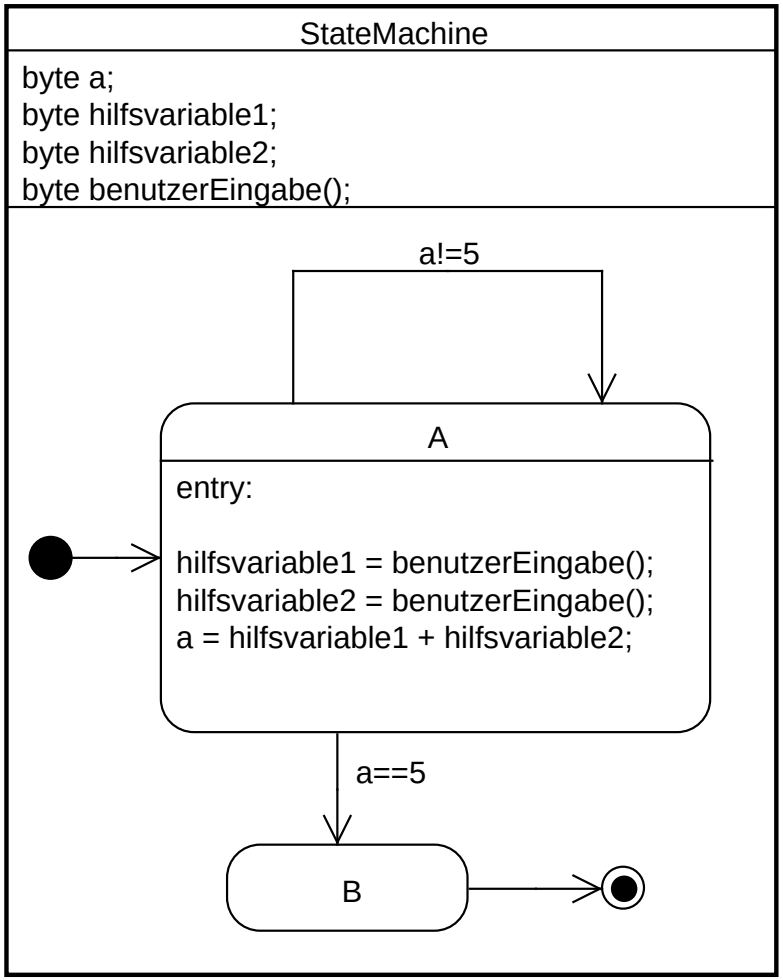
byte a;
byte b;
 	
a=1;
b=2;
b=a*b;
	

Statement Merging

Promela	SPIN Verifier

byte a;
byte b;
 	
d_step{
a=1;
b=2;
b=a*b;
}
	

Zurücksetzen von Hilfsvariablen



Kein Zurücksetzen von Hilfsvariablen

Benutzereingabe: (1,3), (2,2), (3,1)

Gespeicherte Zustände:

- Zustand = A,
  hilfsvariable1 = 1, hilfsvariable2 = 3, a = 4
- Zustand = A,
  hilfsvariable1 = 2, hilfsvariable2 = 2, a = 4
- Zustand = A,
  hilfsvariable1 = 3, hilfsvariable2 = 1, a = 4

Zurücksetzen von Hilfsvariablen

Benutzereingabe: (1,3), (2,2), (3,1)

Gespeicherte Zustände:

- Zustand = A,
  hilfsvariable1 = 0, hilfsvariable2 = 0, a = 4

Messungen bei einer Fallstudie

Problemstellung: Ist es möglich, dass der Modellprüfer den kompletten Zustandsraum durchläuft?

Messung	Speicher	Laufzeit	Zustände	Erfolg
Keine Optimierung	2999 MB	39.2 s	29141706	nein
Statement Merging	2223 MB	33.3 s	20663354	ja
Hilfsvariablen	674 MB	12.8 s	6798196	ja
Symmetrie	3 MB	0.1 s	17087	ja

Fragen