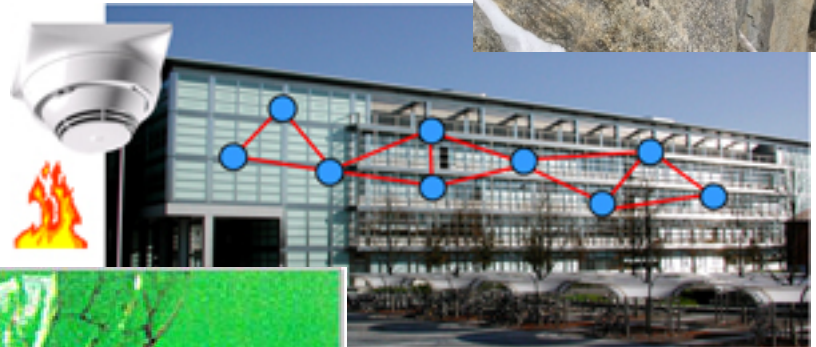
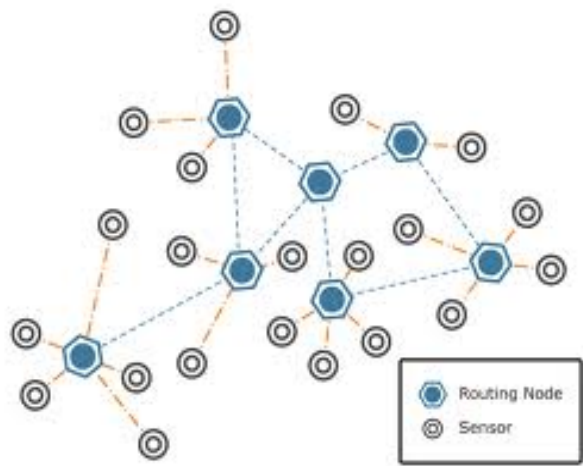


Correct by Construction

proving instead of testing

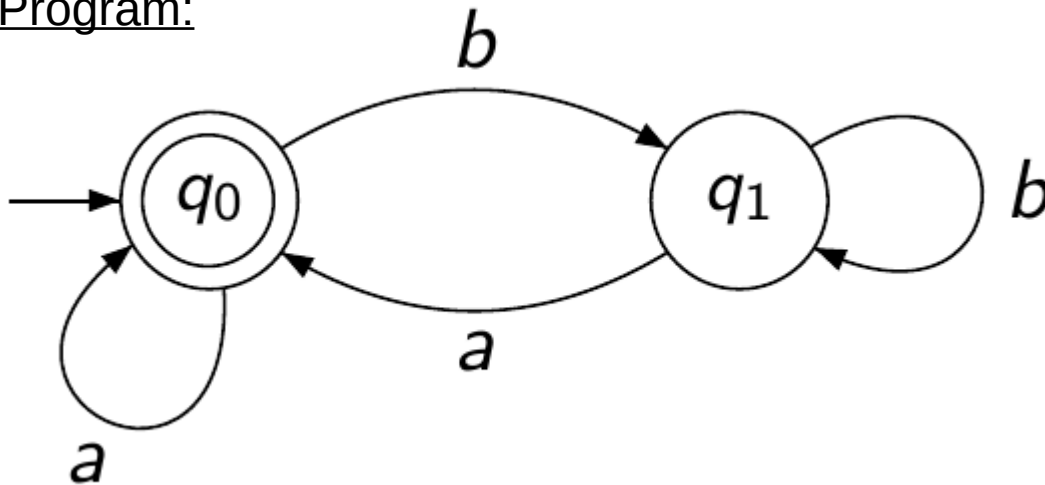
Alexander Bernauer
alex@ulm.ccc.de



Model Checking

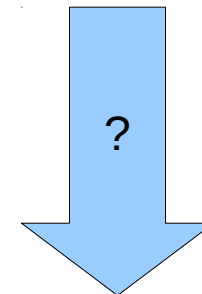
Programm vs. Spezifikation

Program:



Ausführungen:

$q_0 q_1 q_0 q_0 q_1 \dots$
 $q_0 q_0 q_0 q_0 q_0 \dots$
 $q_0 q_1 q_1 q_1 q_0 \dots$
 \dots



Spezifikation:

$\Box \Diamond q_0$

Lineare Temporale Logik:

$\Phi \ \psi$	Prädikatenlogische Formel
$\circ \Phi$	next
$\Box \Phi$	always
$\Diamond \Phi$	eventually
$\psi \ \mathbf{U} \ \Phi$	until
$\psi \ \mathbf{R} \ \Phi$	release

Demo

Vor- und Nachteile

- vollständig
- Gegenbeispiele
- zeitliche Prädikate
- Endlichkeit
- Model vs. Code
- State Explosion

Theorem Proving

Maschine vs. Anforderungen

Konstanten:

`max = 5`

Variablen:

`floor`

Initialisierung:

`floor = 0`

Ereignisse:

up:

guards:

`floor < max`

actions:

`floor = floor + 1`

down:

guards:

`0 < floor`

actions:

`floor = floor - 1`


Invarianten:

`0 <= floor`

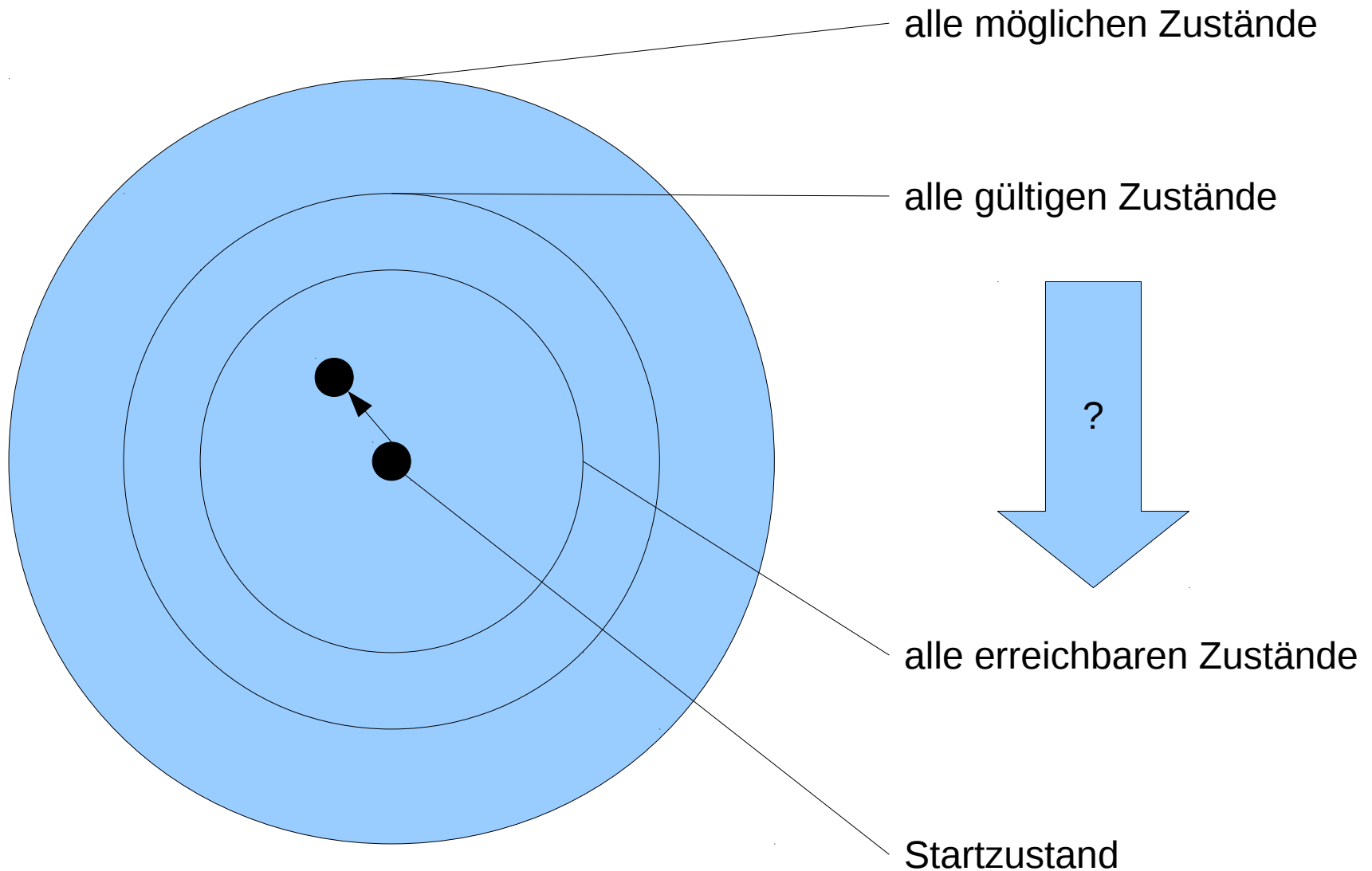
`floor <= max`

Proof Obligation

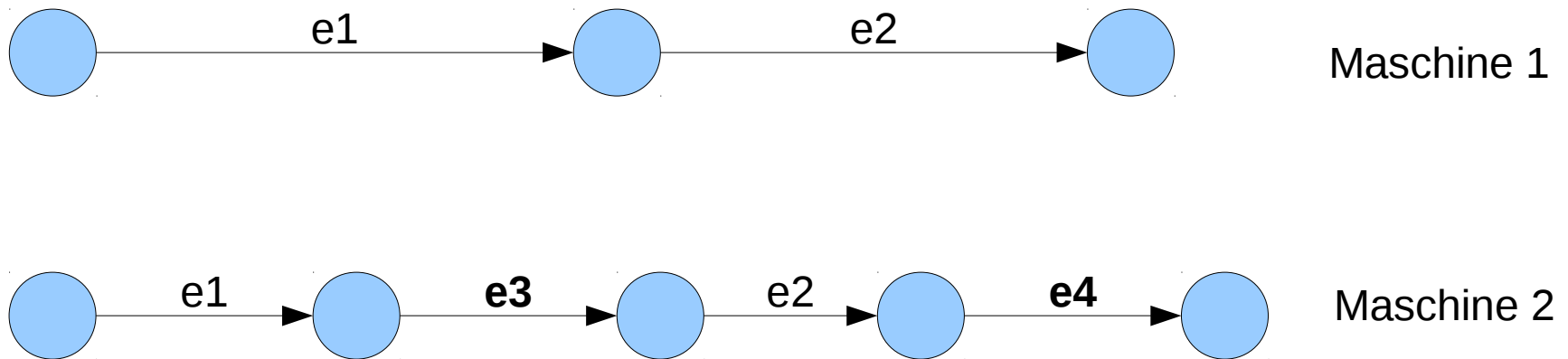
für jedes Ereignis und jede Invariante

Invarianten: $0 \leq \text{floor}$ $\text{floor} \leq \text{max}$	<u>vor dem Ereignis</u> Guards sind wahr Invarianten sind wahr		<u>nach dem Ereignis</u> Invariante ist wahr
up $\text{floor} \leq \text{max}$	$\text{floor} < \text{max}$ $0 \leq \text{floor}$ $\text{floor} \leq \text{max}$		$\text{floor} + 1 \leq \text{max}$
down $0 \leq \text{floor}$	$0 < \text{floor}$ $0 \leq \text{floor}$ $\text{floor} \leq \text{max}$		$0 \leq \text{floor} - 1$
init			$0 \leq 0$ $0 \leq 5$
Initialisierung: $\text{floor} = 0$	down: guards: $0 < \text{floor}$ actions: $\text{floor} = \text{floor} - 1$		up: guards: $\text{floor} < \text{max}$ actions: $\text{floor} = \text{floor} + 1$

Induktive Beweise



Refinements



Proof Obligations:
Guard Strengthening
Action Simulation
Konvergenz

Demo

Vor- und Nachteile

- vollständig
- Unendlichkeit
- korrekter Code
- Akzeptanz
- autom. Beweiser
- Induktivität

Zusammenfassung

	Testing	Model Checking	Theorem Proving
Laufzeit	ja	simuliert	nein
vollständig	×	√	√
unendlich	×	×	√
Fehlersuche	Stacktrace	Gegenbeispiel	unvollst. Beweis
Anwendbarkeit	±	±	±
formal	0	+	++
Akzeptanz	+	0	-
Flexibilität	+	0	0

Danke

- <http://event-b.org>
- <http://www.comp.nus.edu.sg/~pat/>